1267923

THE UNITED STATES OF AMERICA

TO ALL TO WHOM THESE PRESENTS SHALL COME:

UNITED STATES DEPARTMENT OF COMMERCE

United States Patent and Trademark Office

January 03, 2005

THIS IS TO CERTIFY THAT ANNEXED HERETO IS A TRUE COPY FROM THE RECORDS OF THE UNITED STATES PATENT AND TRADEMARK OFFICE OF THOSE PAPERS OF THE BELOW IDENTIFIED PATENT APPLICATION THAT MET THE REQUIREMENTS TO BE GRANTED A FILING DATE.

APPLICATION NUMBER: *60/562,908*
FILING DATE: *April 16, 2004*
RELATED PCT APPLICATION NUMBER: *PCT/US04/37761*

Certified by

[illegible]

Jon W Dudas

Acting Under Secretary of Commerce
for Intellectual Property
and Acting Director of the U.S.
Patent and Trademark Office

UNITED STATES PATENT AND TRADEMARK OFFICE
DEPARTMENT OF COMMERCE

16191 U.S. PTO

13441 U.S. PTO 60/562908 041604

PROVISIONAL APPLICATION FOR PATENT COVER SHEET

	Docket Number	126709.700	Type a plus sign (+) inside this box →	+

INVENTOR(s)/APPLICANT(s)

LAST NAME	FIRST NAME	MIDDLE INITIAL	RESIDENCE (CITY AND EITHER STATE OR FOREIGN COUNTRY)
Horton	James	A.	New Tripoli, Pennsylvania
Gross, Jr.	George	F.	Fleetwood, Pennsylvania
Klein, Jr.	Robert	C.	Macungie, Pennsylvania
Flemming	Terrence	E.	Macungie, Pennsylvania

TITLE OF THE INVENTION (280 characters max)

MODULAR PROCESSING ARCHITECTURE WITH A SELF-ROUTING, MESSAGE-BASED INTERCONNECT SYSTEM

CORRESPONDENCE ADDRESS

Firm ID 21269

Pepper Hamilton LLP
500 Grant Street
One Mellon Center, 50th Floor
Pittsburgh, PA 15219

ENCLOSED APPLICATION PARTS (check all that apply)

[X] Specification (Number of pages) [33] [ ] CDs (number) [ ]
[X] Drawings (Number of sheets) [10] [X] Other (specify): Postcard, Certificate of Mailing, Fee Transmittal and Check

METHOD OF PAYMENT (check one)

[X] Applicant(s) claim(s) small entity status. See 37 C.F.R. § 1.27. [X] A check or money order is enclosed to cover the Provisional filing fees [X] The Commissioner is hereby authorized to charge fees to Deposit Account Number: 50-0436	PROVISIONAL FILING FEE AMOUNT ($)	$80.00

he invention was made by an agency of the United States Government or under a contract with an agency of the United States Government.

x] No [ ] Yes

] Additional inventors are being named on separately numbered sheets attached hereto.

Respectfully submitted,

[illegible]

James M. Singer
Registration No. 45,111

)ate: April 16, 2004

T: #160209 v1 (3V1T01!.DOC)

16191 U.S. PTO

PTO/SB/17 (10-03)
Approved for use through 07/31/2006. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

FEE TRANSMITTAL for FY 2004

Effective 10/01/2003. Patent fees are subject to annual revision.

[✓] Applicant claims small entity status. See 37 CFR 1.27

TOTAL AMOUNT OF PAYMENT ($) 80.00

Complete if Known

Application Number	not yet assigned
Filing Date	April 16, 2004
First Named Inventor	Horton, James, et al.
Examiner Name	not yet assigned
Art Unit	not yet assigned
Attorney Docket No.	126709.700

METHOD OF PAYMENT *(check all that apply)*

[✓] Check [ ] Credit card [ ] Money Order [ ] Other [ ] None

[✓] Deposit Account:

Deposit Account Number

Deposit Account Name

The Director is authorized to: *(check all that apply)*

[✓] Charge fee(s) indicated below [✓] Credit any overpayments

[✓] Charge any additional fee(s) or any underpayment of fee(s)

[ ] Charge fee(s) indicated below, except for the filing fee

to the above-identified deposit account.

FEE CALCULATION

1. BASIC FILING FEE

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
1001	770	2001	385	Utility filing fee	
1002	340	2002	170	Design filing fee	
1003	530	2003	265	Plant filing fee	
1004	770	2004	385	Reissue filing fee	
1005	160	2005	80	Provisional filing fee	80.00

SUBTOTAL (1) ($) 80.00

2. EXTRA CLAIM FEES FOR UTILITY AND REISSUE

			Extra Claims		Fee from below		Fee Paid
Total Claims		-20** =		X		=	
Independent Claims		- 3** =		X		=	
Multiple Dependent						=	

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description
1202	18	2202	9	Claims in excess of 20
1201	86	2201	43	Independent claims in excess of 3
1203	290	2203	145	Multiple dependent claim, if not paid
1204	86	2204	43	** Reissue independent claims over original patent
1205	18	2205	9	** Reissue claims in excess of 20 and over original patent

SUBTOTAL (2) ($) 80.00

**or number previously paid, if greater; For Reissues, see above*

FEE CALCULATION (continued)

3. ADDITIONAL FEES

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
1051	130	2051	65	Surcharge - late filing fee or oath	
1052	50	2052	25	Surcharge - late provisional filing fee or cover sheet	
1053	130	1053	130	Non-English specification	
1812	2,520	1812	2,520	For filing a request for *ex parte* reexamination	
1804	920*	1804	920*	Requesting publication of SIR prior to Examiner action	
1805	1,840*	1805	1,840*	Requesting publication of SIR after Examiner action	
1251	110	2251	55	Extension for reply within first month	
1252	420	2252	210	Extension for reply within second month	
1253	950	2253	475	Extension for reply within third month	
1254	1,480	2254	740	Extension for reply within fourth month	
1255	2,010	2255	1,005	Extension for reply within fifth month	
1401	330	2401	165	Notice of Appeal	
1402	330	2402	165	Filing a brief in support of an appeal	
1403	290	2403	145	Request for oral hearing	
1451	1,510	1451	1,510	Petition to institute a public use proceeding	
1452	110	2452	55	Petition to revive - unavoidable	
1453	1,330	2453	665	Petition to revive - unintentional	
1501	1,330	2501	665	Utility issue fee (or reissue)	
1502	480	2502	240	Design issue fee	
1503	640	2503	320	Plant issue fee	
1460	130	1460	130	Petitions to the Commissioner	
1807	50	1807	50	Processing fee under 37 CFR 1.17(q)	
1806	180	1806	180	Submission of Information Disclosure Stmt	
8021	40	8021	40	Recording each patent assignment per property (times number of properties)	
1809	770	2809	385	Filing a submission after final rejection (37 CFR 1.129(a))	
1810	770	2810	385	For each additional invention to be examined (37 CFR 1.129(b))	
1801	770	2801	385	Request for Continued Examination (RCE)	
1802	900	1802	900	Request for expedited examination of a design application	

Other fee (specify) ______________

*Reduced by Basic Filing Fee Paid

SUBTOTAL (3) ($) 0

SUBMITTED BY

(Complete *(if applicable)*)

Name (Print/Type)	James M. Singer	*Registration No. (Attorney/Agent)*	45,111	*Telephone*	412.454.5000
Signature	[illegible]			*Date*	April 16, 2004

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

CERTIFICATE OF MAILING UNDER 37 C.F.R. § 1.10

APPLICANT:	Horton, James, et al.
TITLE:	Modular Processing Architecture with a Self-Routing, Message-Based Interconnect System
ATTORNEY REF:	126709.700
SERIAL NO.	To be assigned
DATE OF DEPOSIT:	April 16, 2004

I hereby certify that, on the date shown below. This correspondence is being:

MAILING		FACSIMILE	
[X]	deposited with the United States Postal Service with sufficient postage as United States Express Mail in an envelope addressed to:	[ ]	transmitted by facsimile to the Patent and Trademark Office

Mail Stop Provisional Application, Commissioner for Patents, United States Patent and Trademark Office, P.O. Box 1450, Alexandria, VA 22313-1450.

Date: April 16, 2004

Maureen A. Horgan
Signature

Maureen A. Horgan
(type or print name of person certifying)

MODULAR PROCESSOR ARCHITECTURE WITH A SELF-ROUTING, MESSAGE-BASED INTERCONNECT SYSTEM

BACKGROUND

Field of the Invention

The present invention relates generally to an integrated circuit interconnect method and apparatus and more specifically it relates to a modular processor architecture with a self-routing, message-based interconnect system for electrical and electronic devices. In an embodiment, the present invention provides a flexible, efficient, extensible, self-routing and real-time dynamically-optimized means for connecting together functional elements or blocks within a semiconductor device.

Description of the Prior Art

It can be appreciated that integrated circuits (chips or ICs) with various types of interconnect have been in use for years. Typical integrated circuit interconnections can be grouped into three distinct classes of on-chip interconnect. The first class is conventional microprocessor and/or microcontroller bus-type architectures. These conventional microprocessor/microcontroller bus architectures include Von Neumann type architectures – wherein address and data information are multiplexed onto a single set of metal conductors – and Harvard architectures – where separate paths are provided for the data and address signals. The second class of conventional on-chip interconnect is the programmable interconnect found in Field Programmable Gate Arrays (FPGAs) and other Programmable Logic Devices (PLDs). The third class of on-chip interconnect is the highly customized, hard-wired, device-specific wiring that is found in full-custom, Application Specific Integrated Circuits (ASICs).

A problem with conventional integrated circuit interconnect structures is their inability to be adaptable to and dynamically optimized for a range of tasks. Another problem with conventional integrated circuit interconnect structures is their inability to be dynamically *changed* to precisely meet the requirements of the current task or set of tasks at hand and then be modified should the task or tasks change. While the programmable interconnect within FPGA-type devices can in theory be reconfigured, in practice this is rarely done because of the complexity of the tools required and the time lapse ("configuration latency") associated with such changes. Another problem with conventional integrated circuit interconnect structures is that they are not well-suited for a semiconductor architecture wherein the data paths are constantly changing. Such changes are desirable in order to support real-time optimization of the circuitry and interconnect to match the needs of the task or set of tasks at hand. Furthermore, conventional fixed and programmable routing structures do not allow an array of functions – either homogenous or heterogeneous – to be quickly and easily laid out in a device such that the routing structures can be created and controlled simply by placing the functions physically (or logically) next to each other ("tiling").

While the interconnect structures of the prior art may be suitable for the particular purpose to which they address, they are not as suitable for providing a flexible, efficient, self-routing and dynamically optimized means for connecting together functional elements or blocks within a semiconductor device.

In these respects, the modular processor architecture with a self-routing, message-based interconnect system for electrical and/or electronic devices according to am embodiment of the present invention substantially departs from the conventional concepts and designs of the prior art, and in so doing it may provide an apparatus that allows a flexible, efficient, extensible, self-routing and real-time dynamically-optimized means for connecting together functional elements or blocks within a semiconductor device that is modular in approach and therefore requiring no additional circuitry for applications requiring multiple modules.

SUMMARY OF THE INVENTION

In view of the foregoing disadvantages inherent in the known types of conventional integrated circuit and system-level interconnect structures now present in the prior art, in an embodiment the present invention provides a new modular processor architecture with a self-routing, message-based interconnect system for electrical construction wherein the same can be utilized to provide a flexible, efficient, self-routing and real-time dynamically optimized means for connecting together functional elements or blocks within a semiconductor device and/or other electrical or electronic system.

An embodiment of the present invention, which will be described subsequently in greater detail, provides an interconnect system for electrical and/or electronic devices or systems that has many of the advantages of the conventional integrated circuit interconnect structures mentioned heretofore and one or more novel features that result in a new modular processor architecture with a self-routing, message-based interconnect system for electrical and/or electronic devices which is not anticipated, rendered obvious, suggested, or even implied by any of the prior art integrated circuit interconnect types, either alone or in any combination thereof.

To attain this, in an embodiment the present invention includes a series of processing and/or logic elements to be interconnected; a series of primary and secondary groups (busses) of interconnect paths including Local and/or Long Distance busses with separate signal paths for data, address, and/or control signals; and Arbitration and Control Circuitry.

The Processing Element (referred to herein as a Virtual Function Block or VFB) is one of a series or array of elements to be interconnected. It can include one or more of the following components or any combination thereof: Central Processing Units (CPU), Arithmetic Logic Units (ALU); Memory Elements (MEM); Arbitrary Function Generators (ARB); State Machines; Digital Signal Processors (DSP); Analog Signal Processors (ASP); Programmable Logic Devices (PLD including Field Programmable Gate Array (FPGA) and Complex PLD (CPLD)); Input and/or Output (I/O) elements; and/or General Purpose logic. The array of VFBs can either be homogeneous or heterogeneous. The actual type of blocks being interconnected is not what is being considered in the scope of this invention; it is the method for dynamically connecting the blocks together. The Arbitration and Control Circuitry (described below) is also a part of the VFB, as are ports which physically connect the VFBs to the Interconnect Busses described below.

The Interconnect Busses provide the physical connections over which data – including applications data, addressing, control, program, and signaling information – is passed between the VFBs. In a preferred embodiment, the Interconnect Busses are broken into two types; the Local busses and the Long Distance busses. Each VFB has both input and output Local and Long Distance busses for each of four generalized physical directions for a total of eight Local and eight Long Distance busses. The Local busses operate in conjunction with the Long Distance bus structures and provide dedicated, high-bandwidth communications between physically and/or logically neighboring VFBs in the array.

The Arbitration and Control Circuitry may perform three distinct functions. First, it may be responsible for formatting Message requests that originate in a given VFB and forwarding these Messages accordingly. The first decision for any out-going Message may be determining – from address information contained in the Message – if the Message is intended for any one of the four logically adjacent (Local) VFBs. Local Messages can be initiated explicitly by using one of the dedicated Local Message registers sets (one register for Address information and one for Data; one such set for each of the four generalized directions UP, DOWN, LEFT, and RIGHT) or implicitly by writing the Message Address and Data information to the Long Distance Message register set. In the later case, the Arbitration and Control Circuitry determines from the row/column offset information in the Long Distance Address register that the Message is intended for a local "neighbor", automatically formats the Message as a Local Message, and uses the associated, dedicated Local bus circuitry for passing the Message. If the row/column offset information in the Long Distance Address register does not indicate the use of the Local Message resources, the Message is formatted for and passed through to the Long Distance Message circuitry. This allows transparency from a software perspective of where functions are physically placed in the array while insuring the use of the dedicated Local busses whenever possible.

A second function of the Arbitration and Control Circuitry may be to detect incoming Long Distance Messages from other VFBs, determine the availability of a path that would move the incoming Message closer to its destination, and – if more than one such path exists – select one of the available paths (according to the prioritization scheme described later in this disclosure). If such a path is available, the Arbitration and Control Circuitry forwards the Message down that path, adjusting the row/column offset addressing information accordingly. If a path is NOT available, this circuitry rejects the request (preventing the incoming request from selecting this path) and signals this condition to the previous VFB in the path, thereby automatically forcing a different path to be established. In this manner, all potential minimal-length paths (minimal meaning the minimum possible number of row and column "hops") are rapidly interrogated and the first possible successful route discovered. The Arbitration and Control Circuitry may also prioritize and resolve simultaneous requests for service.

A third function of the Arbitration and Control Circuitry may include determining that an incoming Message has reached its destination, determining/signaling the availability of the destination resource requested in the Message, and – if the destination resource is indeed available – placing the Message payload (data) into the requested resource and signaling the completion of this transaction.. If the destination resource requested is not available, this is signaled back through the established path to the source of the request.

The invention may allow multiple, optimized, simultaneous connection paths to be discovered, used, and then released on a real-time, as-needed basis. In an embodiment, the message handling circuitry is completely combinatorial and not dependent upon any system or local clock. Ergo, the combinatorial path-discovery circuitry interrogates potential minimal-length communication paths – exhaustively if necessary – and establishes the first available path at speeds limited only by the intrinsic delays of the physical silicon in which the invention is implemented. This flexible, dynamic and task-dependant interconnect may provide a very high level of interprocessor communication for parallel processing.

Through the use of embodiments of this invention, systems created from an array of VFBs that employ the communication mechanism of the current invention (regardless of any other specific functionality within the individual VFBs) can be developed and software for such systems written before a physical device is designed and fabricated. By simply placing the required VFBs physically (or logically) adjacent to one another in an array, the interconnect structure and communication protocol for inter-VFB communication is established. The concepts of the current invention apply equally to single, integrated silicon devices, multi-chip modules (McMs) and System-on-Chip (SoC) designs, and board-level (discrete packages connected together on a printed circuit board) designs.

There has thus been outlined, rather broadly, the more important features of the invention in order that the detailed description thereof may be better understood, and in order that the present contribution to the art may be better appreciated. There are additional features of the invention that will be described hereinafter.

In this respect, before explaining at least one embodiment of the invention in detail, it is to be understood that the invention is not limited in its application to the details of construction and to the arrangements of the components set forth in the following description or illustrated in the drawings. The invention is capable of other embodiments and of being practiced and carried out in various ways. Also, it is to be understood that the phraseology and terminology employed herein are for the purpose of description and should not be regarded as limiting.

In an embodiment, the present invention provides a modular processor architecture with a self-routing, message-based interconnect system for electrical and/or electronic devices that overcomes the shortcomings of the prior art interconnect systems.

In an embodiment, the present invention provides a modular processor architecture with a self-routing, message-based interconnect system for electrical and/or electronic devices that provides a flexible, efficient, and dynamically optimized means for connecting together functional elements within a semiconductor device.

In an embodiment, the present invention provides a modular processor architecture with a self-routing, message-based interconnect system for electrical and/or electronic devices that provides a means for easily integrating an array of processing and/or other functional elements that employ the current invention by physically locating the such elements adjacent to one another, thereby eliminating the need for custom or programmable routing interconnect in an integrated device, module, or device array.

In an embodiment, the present invention provides a modular processor architecture with a self-routing, message-based interconnect system for electrical and/or electronic devices that allows the data and control paths within an integrated circuit to be quickly and dynamically modified – in real time and under software control – to precisely match the requirements of the current task or set of tasks that the device is to perform.

In an embodiment, the present invention provides a modular processor architecture with a self-routing, message-based interconnect system for electrical and/or electronic devices that minimizes contention for resources and automatically attempts to resolve any such contention.

In an embodiment, the present invention provides a modular processor architecture with a self-routing, message-based interconnect system for electrical and/or electronic devices that allows multiple connection paths for multiple, concurrent connections to be quickly and automatically discovered, selected, used as needed, and then released.

In an embodiment, the present invention provides a modular processor architecture with a self-routing, message-based interconnect system for electrical and/or electronic devices that dynamically varies the path-selection decision tree thereby eliminating hardware deadlock; software deadlock is still possible based on how such software employs the interconnect system, however, the hardware – by design – will never become deadlocked.

In an embodiment, the present invention provides a modular processor architecture with a self-routing, message-based interconnect system for electrical and/or electronic devices that automatically releases resources after a transfer is complete to allow these resources to be used by other tasks if needed.

In an embodiment, the present invention provides a modular processor architecture with a self-routing, message-based interconnect system for electrical and/or electronic devices that automatically selects paths with minimal physical lengths and propagation delays.

In an embodiment, the present invention provides a modular processor architecture with a self-routing, message-based interconnect system for electrical and/or electronic devices that dynamically creates interconnection paths between sources and destinations wherein the timing of the interconnect between any individual source and destination is invariant regardless of the path selected.

In an embodiment, the present invention provides a modular processor architecture with a self-routing, message-based interconnect system for electrical and/or electronic devices that dynamically creates interconnection paths between sources and destinations that can operate in unrelated clock domains.

In an embodiment, the present invention provides a modular processor architecture with a self-routing, message-based interconnect system for electrical and/or electronic devices that can synchronize processing elements and/or logic operating in unrelated clock domains.

In an embodiment, the present invention provides a modular processor architecture with a self-routing, message-based interconnect system for electrical and/or electronic devices that asynchronously discovers and establishes non-qued, non-pipelined interconnections paths, the speed of which are limited only by the delay of the combinatorial logic in each path and not limited by any clocking mechanism.

In an embodiment, the present invention provides a modular processor architecture with a self-routing, message-based interconnect system for electrical and/or electronic devices that automatically senses blocked paths, and – if so blocked –

automatically and, if necessary, exhaustively searches all alternate, equally beneficial paths.

Other objects and advantages of the present invention will become obvious to the reader and it is intended that these objects and advantages are within the scope of the present invention.

To the accomplishment of the above and related objects, this invention may be embodied in the form illustrated in the accompanying drawings, attention being called to the fact, however, that the drawings are illustrative only, and that changes may be made in the specific construction illustrated.

BRIEF DESCRIPTION OF THE DRAWINGS

Various other objects, features and advantages of the present invention will become fully appreciated as the same becomes better understood when considered in conjunction with the accompanying drawings, in which like reference characters designate the same or similar parts throughout the several views, and wherein:

Figure 1 is a block diagram of an exemplary Messenger Interconnected Processor Array.

Figure 2 provides a block diagram of an Individual VFB.

Figure 3 illustrates the various sub-blocks that may combine to form an individual VFB, and the signals that may interconnect these blocks.

Figure 4 illustrates a Local Messenger Port.

Figure 5 illustrates a Local Messenger.

Figure 6 illustrates a Long Distance Messenger Port.

Figure 7 shows the Long Distance Messenger Port Control Logic.

Figure 8 illustrates the Long Distance Messenger.

Figure 9 illustrates how Request and Busy Information may be shared among Long Distance Port Controls.

Figure 10 shows a Long-Distance Message Connection with no path contention.

Figure 11 illustrates A Long Distance Message Connection, but with path contention and without Deny.

Figure 12 shows a Long Distance Message Connection, but with Path Contention and with Deny.

DETAILED DESCRIPTION

Throughout the figures provided, numbering is preserved such that a reference number appearing in more than one figure refers to the same object. Arrows within the figures refer to the primary direction of the flow of control and data and should not be construed as the sole direction of data flow.

Referring to Figure 1, in an embodiment the invention includes a two-dimensional array (100) of processing elements, referred to as Virtual Function Blocks or VFBs (101), interconnected by message passing ports (102) that provide connections between adjacent VFBs through two distinct, bi-directional paths, the local and long distance connection paths. The term adjacent, throughout this description, refers to the connection relationship between VFBs and not necessarily their physical orientation to each other. For example, interleaving the local and long distance connections between VFBs such that the connections pass over physically adjacent VFBs results in VFBs that are not physically adjacent being connected adjacently for the purpose of this description. For simplicity in this description, adjacent VFBs are also physically adjacent VFBs.

Refer to Figure 2, a more detailed view of a VFB (101) and its connections, for the following descriptions. Local messenger ports connect between adjacent VFBs in the right (201), down (202), left (203) and up (204) directions. Similarly, long distance messenger ports connect between adjacent VFBs in the right (205), down (206), left (207) and up (208) directions. The local messenger ports are used to relay messages between adjacent source and destination VFBs in any of the orthogonal directions; up, down, left or right. The long distance messenger paths are similarly connected between adjacent VFBs, but contain additional addressing information that is used and manipulated by each VFB to construct a path spanning multiple VFBs. Such multi-VFB paths dynamically create a connection between source and destination VFBs within the array that are non-adjacent. A source VFB is considered to be a sender of a message containing control or data to be stored in a destination VFB. The location of the destination VFB and the storage location of the message contents within the destination VFB are controlled by the source VFB. Local messages are sent and received by a local messenger unit (209) using the local messenger ports (201, 202, 203 and 204) contained within a VFB. Long distance messages are sent and received by a long distance messenger unit (210) using the long distance messenger ports (205, 206, 207 and 208) contained within a VFB. Both the local messenger (209) and long distance messenger (210) are contained within the VFB internals (211). Each VFB (101) is specifically designed to be modular, so that no additional logic is required to connect adjacent VFBs together, permitting larger or smaller arrays of VFBs to be readily constructed.

Referring to Figure 3, within the VFB internals (211) is a messenger register file (301), constructed as an array of discrete registers that serves as a portal through which messages are sent to and received by adjacent or non-adjacent VFBs. The register file is connected via a bus arbitrator (302), to a processor (303) that may consist of a CPU, DSP, FPGA, mask PGA and/or any other logic element capable of accessing the registers contained within the messenger register file (301). The register file is also connected to control logic (304) that has continuous access to the messenger register file (301), independently from the bus arbitrator (302). The bus arbitrator (302) and the control logic (304) are capable of accessing the register file concurrently, including when accessing the same register.

Each register within the register file (301) has associated with it a presence bit that indicates the presence (asserted) or absence (not asserted) of data or control information within the register. Each register may be accessed and manipulated in one of three fundamental ways; constructively, passively or destructively. A constructive access is one in which the register is filled with control or data information, causing the presence bit associated with the register to be asserted. A passive access is one in which the contents of the register is inspected but the presence bit associated with the register is left unchanged. A destructive access is one in which the register contents are inspected and the presence bit associated with the register is caused to be not asserted. For any access to a register within the register file, the register file presents the state of the presence bit associated with the register to the accessor so that a signal causing the accessor to wait (halt), suspend or retry may be generated if the access is inhibited.

A constructive access to a register within the register file is inhibited if the presence bit associated with the register is currently asserted. A passive or destructive access to a register within the register file is inhibited if the presence bit associated with the register is currently cleared. The preferred action of the processor (303) to an inhibited access to a messenger register (301) is to wait.

The processes of permitting the processor (303) to wait while still permitting constructive access to the register file by the messages received through the message passing fabric is accomplished by a memory bus arbitrator (302). The memory bus arbitrator selects between competing accesses from the processor (303) and incoming data from the message-passing fabric arriving through the local messenger (305). The memory bus arbitrator gives priority to the local messenger (305), causing the next available messenger register file access to be granted to the local messenger, while concurrently providing a wait signal to the processor (303). In the event that the processor (303) is already engaged in an inhibited access to the register file when the local messenger (305) is also in need of a register file access, the memory bus arbitrator continues to present a wait signal to the processor while transferring control of the address and Data lines to the register file (301) to the local messenger (305). Once the local messenger has completed its access to the messenger file, control of the address and data lines is returned to the processor (303). An access to the messenger register file (301) via the bus arbitrator (302) by the local messenger (305) is never inhibited, thus the local messenger, normally, will not begin an access to the register file unless it has confirmed in advance of requesting the access, via control logic (304), that the access will not be inhibited.

The following is a detailed listing of the individual registers that collective form the messenger register file (302) within a VFB, and exemplary addressing within the processor (303) memory space.

Messenger Register	Base Address (Hex)	Register Index (Hex)	Description
CONFIG	10000	0	VFB Config Info
FAULT	10004	1	Crypto Fault Actions
PKEY	10008	2	Crypto Key Register
LOCATOR	1000c	3	Locator VFB address
DLINK	10010	4	Crypto Decrypt Link Register
DECRYPT	10014	5	Crypto Decrypt Register
ELINK	10018	6	Crypto Encrypt Link Register
ENCRYPT	1001c	7	Crypto Encrypt Register
VMSG	10020	8	VMSG Vector Register
M1..M15	10024	9..17	General Purpose Messages
Reserved	10060-78	18..1e	M Register Expansion
MPRES	1007c	1f	Message Presence Bits
MSENDA	10080	20	Message Send Address
MDATAW	10084	21	Message Send Data
MDATAR	10088	22	Message Rcv Data
Reserved	1008c	23	Reserved
LRIGHTA	10090	24	Local Message Send Right Reg Addr
LRIGHTW	10094	25	Local Message Send Right Data
LDOWNA	10098	26	Local Message Send Down Reg Addr
LDOWNW	1009c	27	Local Message Send Down Data
LLEFTA	100a0	28	Local Message Send Left Reg Addr
LLEFTW	100a4	29	Local Message Send Left Data
LUPA	100a8	2a	Local Message Send Up Reg Addr
LUPW	100ac	2b	Local Message Send Up Data
Reserved	100b0-f8	2c..3e	Reserved
MSTATUS	100fc	3f	Messenger Status Bits

Table 1. Messenger Register File

Messenger Register Descriptions

The messenger registers from 10000 hex to 100ff hex are shadowed (not physically) at addresses 10100 hex to 101ff hex. When a register is read in the address range 10000 hex to 100ff hex, the access is passive as described previously. When a register is read in the addresses range 10100 hex to 101ff hex, the access is destructive, causing the presence bit of the register to be cleared. All write accesses to the messenger register file are constructive.

Referring to Table 1, the following is an overview of the general structure and function of selected individual registers within the messenger register file that are pertinent to the description of the preferred embodiment.

MPRES

Referring to Table 1, each messenger register within the register index range of 0 to 1f hex has the status of its associated presence bit reflected in the 32 bit contents of the MPRES register. There is a direct correlation between register index value (Table 1) and bit position within the MPRES register, with register index 0 represented by the least significant bit of MPRES and register index 1f hex represented by the most significant bit of MPRES. Access to the MPRES register, regardless of type (constructive, passive or destructive), never results in the access being inhibited. During a constructive access, asserting any bit position of this register causes the corresponding presence bit it represents to be not asserted.

MSENDA

Referring to Table 2, the MSENDA register contains a description of where a source VFB is requesting to send a message, including the location within the destination where the contents of the message is to be placed. The Register Index field describes the messenger register into which the message data is to be placed. Row Offset, Column Offset, Row Negative and Column Negative combined describe the relative position of the destination VFB relative to this VFB. A row is considered a left-right sequence of adjacent VFBs within the array (100) and a column an up-down sequence of adjacent VFBs within the array (100). Interrupt is used as an optional means to signal the destination VFB upon the arrival of message data from the source VFB. Force, also optional, provides a means for the source VFB to signal the destination VFB to ignore the presence bit of the desired register within the destination VFB, thus allow a transfer that would might otherwise be inhibited to be performed. Burst causes a source and destination VFB to remain connected after the first transfer, rather than causing the connection to be re-established for each transmission of message data.

Name	**Bits**	**Description**
Register Index	0:5	Register index within the destination VFB
Column Negative	6:6	Destination VFB relative column direction, not asserted for right, asserted for left
Row Negative	7:7	Destination VFB relative row direction, not asserted for down, asserted for up
Column Offset	8:15	Two's complement of the absoluted destination VFB col. offset
Row Offset	16:23	Two's complement of the absoluted destination VFB row offset
Interrupt	24:24	If asserted, interrupts the destination processor after transfer
Force	25:25	If asserted, ignores the state of the presence bit associated with the destination VFB Register
Burst	26:26	If asserted, remains connected after the first transfer
Reserved	27:31	Not used

Table 2. MSENDA Register Bit-field Definitions

MDATAW

MDATAW contains the data to be placed into the destination VFB register indicated by the contents of the MSENDA register.

LRIGHTA

Referring to Table 3, the LRIGHTA register contains of the location with the destination where the contents of the message is to be placed. Its structure and field meanings are identical the MSENDA register except that is does not contain row and column address information because it is only used to describe a transfer to the destination VFB that is adjacent to the right, using the local messenger. Burst is also not used, as the local messenger has a dedicated local messenger connection to the right and thus there is no connection path to dynamically construct. LDOWNA, LLEFTA and LUPA are identical in function and differ only with regard to the direction of the destination.

Bits	Description
0:5	Messenger space register index of peer VFB
6:23	Ignored
24:24	1=Interrupt
25:25	1=Force
26:31	Ignored, must be zero

Table 3. LRIGHTA Register Bit-field Definitions

LRIGHTW

LRIGHTW contains the data to be placed into the destination VFB register indicated by the contents of LRIGHTA. LDOWNW, LLEFTW and LUPW are identical in function and differ only with regard to the direction of the destination.

MSTATUS

Referring to Table 1, each messenger register within the register index range of 20 hex to 3f hex, has the status of its associated presence bit reflected in the 32 bit contents of the MSTATUS register. There is a direct correlation between register index value (Table 1) and bit position within the MSTATUS register, with register index 20 hex represented by the least significant bit of MSTATUS and register index 3f hex represented by the most significant bit of MSTATUS. Access to the MSTATUS register, regardless of type (constructive, passive or destructive), never results in the access being inhibited. During a constructive access, asserting any bit position of this register causes the corresponding presence bit it represents to be not asserted.

Local Message Transmission

Referring again to Figure 3, the process of sending a local message from a source VFB to a destination VFB – in this example an adjacent VFB to the right – begins with the processor (303) of the source VFB performing a constructive write access to local messenger register LRIGHTA in the messenger register file (301). The register index portion of LRIGHTA indicates the specific register to be written within the adjacent VFB to the right. This is followed by the processor (303) performing a constructive write access to local messenger register LRIGHTW with the control or data to be transferred into the destination VFB. As a result of these two constructive accesses, the presence bits associated with the LRIGHTA and LRIGHTW registers have been asserted. In the preferred embodiment, performing a constructive access to the LRIGHTW register also causes the presence bit associated with the LRIGHTA register to be asserted concurrently. This is done to facilitate repetitive message transfers without having to perform an additional constructive access to the LRIGHTA register for each transmission.

The control logic (304) of the source VFB, upon detecting that the presence bits associated with the LRIGHTA and LRIGHTW registers are asserted, transfers the Register Index, Force and Interrupt portions of the LRIGHTA register to the local messenger port (201) that exits the right side of the source VFB and enters the left side of the destination VFB (203). Referring to Figure 4, the signals of a local messenger port are shown in greater detail. Thus messenger port (400) is a more detailed view of each of local messenger ports (310, 201, 202, 203 and 204). The signals of these port consist of a Req line, Register Index bus, a Presence line, Force line, Interrupt line, Data bus and Ack line. One set of these lines is fed to an adjacent VFB by an output port (402) and the other identical set is fed from the same adjacent VFB into an input port (401). Thus, each port is a bi-directional connection to an adjacent VFB that utilizes discrete paths. A source VFB will transfer out of the VFB through the output port (402) that feeds in the direction of the destination VFB, to a destination VFB that transfers in using the input port (401) fed from the direction of the source VFB. In this example – a local transmission to the right – the source VFB uses the local port (201) to communicate with the local port (203) of the destination VFB.

Referring to the local messenger port (400), Req is driven by a source VFB, and while asserted, the Register Index, Interrupt and Force lines of the port are assured valid and propagated to the adjacent VFB. Req remains asserted until the destination VFB has completed the transfer, signaled by the assertion of the Ack line by the destination VFB. When the source VFB is not asserting Req, the destination VFB causes Ack to be not asserted.

Also referring to the local messenger port (400), Register Index is driven by the source VFB to indicate to the destination VFB the messenger register to which a constructive access is required. Presence is driven by the destination VFB to indicate the state of the presence bit associated with the messenger register within the destination VFB as indicated by Register Index. The Register Index value from the input port is relayed to control logic (304), connected to the messenger register file (301). The control logic within the destination VFB continuously relays the state of the presence bit associated with the Register Index value through the local messenger (305) to the to input port. The state of the presence bit is continuously reflected, even in the absence of the assertion of Req, thus is independent of the port selection made by the local messenger (500).

Also referring to the local messenger port (400), Force is driven by the source VFB while Req is asserted to indicate to the destination VFB that a constructive transfer is to be performed even if the access to the messenger register within the destination would otherwise be inhibited due to the presence bit associated with that register being currently asserted. Interrupt is driven by the source VFB while Req is asserted to indicate to the destination VFB that, upon the completion of the constructive access to the messenger register within the destination, the processor within the destination VFB is to be signaled by another means additional to the asserting of the presence bit. An example of this is the assertion of a hardware interrupt line if the processor (303) within the destination VFB were a CPU.

Also referring to the local messenger port (400), Data is a 32 bit data or control value driven by the source VFB and received by the destination VFB. Ack is asserted by the destination VFB to signal the source VFB that the requested transfer has been performed. Whenever Req is not asserted, the destination VFB reacts by causing Ack to be not asserted. Once Req has been asserted by the source VFB, Ack will be asserted by the destination VFB only when the transfer into the destination register file has completed. The source VFB responds to the assertion of Ack by causing Req to be not asserted.

Each local messenger input port (401) within the destination VFB is continuously provided with the state of the presence bit of the messenger register specified by the Register Index portion of the input port, via local messenger (305) and control logic (304). The state of the presence bit is provided without regard to the state of the Req line. When the presence bit is indicated as not asserted, or its state overridden by the Force input of the local messenger port, and the Req line of the input port is asserted, the local messenger input port asserts a service request to the priority encoder (500) within the local messenger (305).

Referring to Figure 5, the local messenger (305) within the destination VFB contains a five input priority encoder and source selector (500) that chooses one from the five, potentially concurrent, local requests that may arrive into a destination VFB via the local messenger input ports (201, 202, 203, 204 and 310). Four of these request sources are the local messenger input ports driven by the local messenger outputs ports of adjacent VFBs to the right, below, left and up of the destination VFB. The fifth input (310) is fed from a bridge (311) between the long distance messenger (313) and the local messenger system (305). This bridge is described in detail later on in this disclosure. In the current example, the destination VFB receives the local message request from the source VFB right output port (201), into the left local messenger input port (203) of the destination VFB.

In deciding between concurrent local messenger requests arriving at the local messenger input ports of the VFB, the priority encoder (500) within the destination VFB gives preference to the long distance bridge (310), specifically to minimize the delay associated with long distance messages. The priority of selection for the remaining four local messenger input ports is arbitrary.

Once the priority encoder (500) selects a local message input source to process, requests from all other local messenger input sources within the VFB are locked out until the completion of the selected request. The priority encoder (500) then causes the source selector to generate an address referencing the register described by the register index value present on the selected local messenger input port, in the current example the left port (203), and asserts the generated address and the data from the selected input port to the memory bus arbitrator (302) in the form of a constructive access.

The bus arbitrator (302), at the next available opportunity, gains access to the messenger register file (301) and performs the constructive access. Once the access is completed, the source selector (500) signals the completion of the transfer to the currently selected local messenger input port by asserting the Ack line of the input port, and, if the Interrupt input of the selected input port is asserted, also causes an interrupt to be asserted to the processor (303) within the destination VFB.

The local messenger output port of the source VFB (201) in turn responds to the assertion of Ack at the destination VFB port (203) by de-asserting Req. This is in turn detected by the local messenger (305) which causes the control logic (304) of the source VFB to de-assert the presence bits of both the LRIGHTA and LRIGHTW registers.

The operation of each local messenger input port (201, 202, 203 and 204), including the input port provided from the long distance messenger bridge (310), is identical. Further, the above example of a local message sent from a source VFB to the adjacent VFB destination to its right extends to all of the local message directions, with only the port directions and source VFB message registers changing according to the direction desired. Table 4 describes the source VFB registers, source VFB local message output ports, and destination VFB local message input ports used for each potential local message transfer from a source VFB. Thus any two adjacent VFBs may perform the role of source or destination. Further, sufficient resources are present to permit a VFB to be both a source and destination, concurrently, with all of its adjacent VFBs.

Source VFB Message Registers	**Source VFB Local Messenger Output Port**	**Destination VFB Local Messenger Input Port**
LRIGHTA, LRIGHTW	Right side	Left side
LDOWNA, LDOWNW	Down side	Up side
LLEFTA, LLEFTW	Left side	Right side
LUPA, LUPW	Up side	Down side

Table 4. Relationship of Source VFB Local Message Registers to Source VFB Output Ports and Destination VFB Input Ports

In an embodiment, an important attribute of the message handling circuitry is that it m ay be completely combinatorial and not dependent upon any system or local clock. Ergo, in an embodiment the circuitry interrogates, discovers, uses and then releases communications paths at speeds limited only by the intrinsic delays of the physical silicon in which the invention is implemented.

Long Distance Messenger

Referring to Figure 3, a communications mechanism between non-adjacent VFBs, the long distance messenger (313), may also be provided. While it interconnects adjacent VFBs in a manner similar to the local messenger system (305), its function is markedly different in that is it is used to construct the equivalent of a local messenger path between non-adjacent VFBs by using the long distance messenger ports between each VFB as individual segments of a combined path to the desired destination VFB. To do so, it constructs a segmented path through multiple VFBs by using a relative row and column offset address provided by a source VFB to establish a connection to the destination.

The basic operation of the long distance message transmission is that a source VFB performs constructive accesses to its MSENDA and MDATAW registers, with the contents of MSENDA specifying a relative offset to a VFB destination that is not adjacent, and a register within the destination to be modified. The long distance messenger, using the destination VFB offset, then selects a long distance output port in the direction, either horizontal (column) or vertical (row), towards the destination through the array of VFBs and creates a connection to the port. Each output port causes the relative offset fed out if its port to its adjacent VFB to be decreased according to its direction. Thus, as each connection made through a VFB, the path moves closer to the destination, continuing to decrease the offset as it passes through VFBs until the destination VFB is reached with a zero offset. A final connection is then made to an internal long distance output port that feeds a bridge within the destination. The bridge then converts the long distance request into a local messenger request.

Upon receiving the request, the destination asserts a Grant line back through the established path to the source VFB indicating it is ready to accept the data. The source VFB then feeds the data onto the connection and asserts a strobe line. Upon receiving strobe, the destination VFB causes the data to be placed into the desired messenger register, then causes the Grant line to become not asserted. Once the source VFB receives the non asserted Grant line and has no further data to transfer, it ceases to assert a long distance messenger request. This causes each path connection made between the source VFB and destination VFB to be released.

If, when the initial connection is made to the destination VFB, the register where the data is to be placed is unavailable, an additional line within the connected path, Abandon, is used by the destination VFB to signal the source VFB that it is unable to perform the transfer, and causes the source VFB to abandon the request and release the path. The source will then reattempt the transfer after a delay.

In the process of establishing a path between the source and destination, portions of the desired path may already be consumed by separate connections made between different source and destination pairs. The long distance messenger, upon detecting a blocked path, will chose another port if possible. If no paths exist within the VFB, the long distance messenger asserts a Deny signal to advise the previous VFB in the path that it is unable to forward the connection. The Deny signal causes the previous VFB to select an alternate path in an attempt to circumvent the blockage. Using this mechanism, all minimal-length paths between the source and destination will be attempted until a path is discovered. If no path is available, Deny will be asserted to the source VFB.

Once connected, either a single data value or multiple data values may be sent from the source VFB to the destination VFB, with the Grant signal performing a flow control function for multiple transfers. The source VFB may also elect to construct the path and leave it in a Connected State for an unlimited time, so as it keep a minimum delay connection between it and the destination VFB.

Each VFB contains four long distance messenger ports (205, 206, 207 and 208), each of which is connected to a corresponding long distance messenger port of each of its adjacent VFBs to the right, down, left and up. An additional long distance messenger output port, the long distance messenger internal output (312) is fed to the long distance to local bridge (311), which converts a long distance message request into a local message request that feeds the local messenger internal input port (310). An additional long distance messenger input port; the long distance messenger internal input (314), is fed by control logic (304), and is the means through which a source VFB introduces a message request into the long distance messenger interconnect. The internal input port (314) and internal output port (312) in combination form a fifth long distance messenger port, the internal port.

The transmission of a long distance message is initiated by a processor (303) performing a constructive access to the MSENDA register of the messenger register file (301) within the source VFB. The contents of the MSENDA register specify the register index within the destination VFB where the message data is to be placed, and the relative location of the destination VFB from the source destination and additional operational flags described in detail later in this document. Next, the processor (303) in the source VFB performs a constructive access to the MDATAW register of messenger register file (301), providing the data or control information to be transferred to the specified message register in the destination VFB.

The control logic (304) of the source VFB, upon detecting that the presence bits of the MSENDA and MDATAW registers have been asserted as a result of the constructive accesses, indicates the presence of a long distance request to a path optimizer (315). The control logic (304) provides the optimizer with the address information present in the MSENDA register to determine which mechanism, local messenger or long distance messenger, should perform the transfer. If the relative address indicated in MSENDA specifies an adjacent VFB, i.e. the combined row and column offsets indicates a relative row of zero and column +/- 1, or a relative column of zero and row +/- 1, then the message request is transferred into the appropriate local messenger registers as indicated in Table 5. This is accomplished by the control logic (304) performing a destructive access to the MSENDA register and transferring its contents via constructive access into the local messenger address register that corresponds to the row/column offset described above. Similarly, the control logic performs a destructive access to the MDATAW register and transfers via constructive access its contents into the local messenger data register for the corresponding direction. This transfer is performed independently of the memory bus arbitrator (302). For any combination of row and column offsets not specifically listed in Table 5, the long distance messenger is utilized.

Relative Row Address	Relative Column Address	Local Messenger Registers
+1	0	LDOWNA, LDOWNW
-1	0	LUPA, LUPW
0	+1	LRIGHTA, LRIGHTW
0	-1	LLEFTA, LLEFTW

Table 5. Source VFB Row/Column Relative Address Pairs in MSENDA that Result in Local Messages and the Associated Local Messenger Registers

Referring to Figure 6, each long distance port (600) consists of an input port (601) and output port (602). Thus messenger port (600) is a more detailed view of each of the long distance messenger ports (205, 206, 207 and 208) and the internal messenger port consisting of the internal input port (314) and internal output port (312). Req, when asserted to an input port, causes the Address, Strobe and Data lines to be acted upon by the input port.

Also referring to Figure 6, Address is asserted to an input port by an output port, and consists of relative row and column offset expressed in the form of a relative row offset, a relative column offset, a negative row direction bit and a negative column direction bit. All offset values are expressed as the twos complement value of the absolute relative offset. When the row negative bit is asserted, the row direction of message travel is up; when not asserted, the direction is down. When the column negative bit is asserted, the column direction of message travel is left; when not asserted, the direction is right. When exiting through a left or right port, the relative column portion of the address is decreased. When exiting through an up or down port, the relative row portion of the address is decreased. Because the relative row and column offsets are expressed as the twos complement of the absolute offset, decreasing the offset, in this context, is accomplished by adding one to the row or column offset value.

Also referring to Figure 6, Data is asserted to an input port by an output port, and consists of thirty two signal lines that contain control and data information to be passed unchanged through the long distance messenger to a selected output port. Strobe is asserted to an input port by an output port and passes unchanged to a selected output port. Strobe is driven by the source VFB concurrently with valid data to signal the destination VFB of a requested constructive register access into the destination VFB register file.

Grant is asserted to a selected output port by its adjacently connected input port, and passes unchanged through the long distance messenger to the input port selecting the output port. When asserted, it signals the ability of a destination VFB to perform a constructive access to its register file.

Abandon is asserted to a selected output port by its adjacently connected input port, and passes unchanged through the long distance messenger to the input port selecting the output port. It is asserted by the destination VFB to signal the source VFB that the destination is unable to perform the transfer because the presence bit of the desired destination register currently has its presence bit asserted.

Deny is asserted to a selected output port by its adjacently connected input port to signal the lack of availability of a path segment, or group of segments, and is used to cause the selection of an alternative message path if available.

Disconnected State

The source VFB is considered to be in the disconnected state when the Req line of the long distance messenger internal input (314) is not asserted.

Connecting State

If the optimizer (315) has determined that the VFB address of the message to be sent by the source VFB requires the use of the long distance messenger, the control logic (304) causes the Connecting State to be entered by causing the contents of the source VFB MSENDA register to be placed onto the Data lines of the long distance messenger internal input (314). The control logic (304) also places the row offset, column offset, negative row bit and negative column bit from the MSENDA register contents into the corresponding bit positions of the address portion of the long distance messenger internal input (314). The control logic (304) then asserts the Req line of the long distance messenger internal input.

Long Distance Path Establishment

Referring to Figure 8, each long distance messenger port (205, 206, 207 and 208), and the long distance internal port, has an instance of port control logic (801, 802, 803, 804 and 805) associated with it. Each instance of the port control logic is connected to an associated long distance messenger port, and relays the input port portion of that port to an individual distribution bus (808) to each of the other port controls. Each port control has a selector that is capable of selecting from one of the distribution busses, and passing the information on the bus to the output portion of the messenger port to which it is connected. Thus, within the long distance messenger, any long distance input port may be forwarded to any other long distance output port. A status bus (806) contains request and busy information for each port control so that each port control is able to make a decision based on the desired direction of travel given the ports that are not currently busy. Referring to Figure 9, each port control has an individual request line and individual busy line for each possible direction. For example, a transfer through the right port control (801), can be requested by the internal port control (805), the down port control (802), the left port control (803) or the up port control (804), and supplies a busy indicator line to each.

Referring to Figure 7, a more detailed view of the port logic is shown. The port control logic has two components, input control (603) and output control (604). The input control (603) is connected to a long distance messenger input port (601) for an individual direction, and the output control (604) is connected to a long distance messenger output port (602) for the same direction. The input control is responsible for selecting an output port within the long distance messenger that is capable of moving the message data towards the destination VFB. The output control, based on requests made by the input controls, grants access to an input port by selecting its data from one of the distribution busses (808) and thus relays the selected input port to its output.

Referring to Figure 7, when Req is not asserted to a long distance messenger input port, the input port control logic (603) forces Grant, Deny and Abandon to be not asserted. The input port control logic maintains latched state information consisting of row inhibit (701) and column inhibit (702) flags that are caused to be not asserted when the Req line of the input port is not asserted. The row inhibit is asserted when it is determined that the up and down output port directions are either in use or of no use given the desired direction of message travel indicated to the port input. The column inhibit is asserted when it is determined that the left or right output port directions are either in use or of no use given the desired direction of message travel indicated to the port input.

When Req is asserted to the input port, the address inspection block (703) inspects the address information present on the input port. If the row offset is zero, the row inhibit (701) of the port control is asserted. If the column offset is zero, the column inhibit (702) of the port control is asserted.

The direction selector (704), based on the address values fed through the address selection block (703), generates one or more potential output port choices according to Table 6.

Row Negative Direction Bit	**Row Address Offset**	**Column Negative Direction Bit**	**Column Address Offset**	**Potential Output Port Choices**
Don't Care	Zero	Don't Care	Zero	Internal
False	Non Zero	Don't Care	Zero	Down
True	Non Zero	Don't Care	Zero	Up
Don't Care	Zero	False	Non Zero	Right
Don't Care	Zero	True	Non Zero	Left
False	Non Zero	False	Non Zero	Down, Right
False	Non Zero	True	Non Zero	Down, Left
True	Non Zero	False	Non Zero	Up, Right
True	Non Zero	True	Non Zero	Up, Left

Table 6. Direction Selector: Potential Output Port Choices Based on Address Values Fed Through the Address Selection Block

The potential output port choices are further qualified within the direction selector (704) by combining the potential output port choices with the state of row inhibit (701) and column inhibit (702). Any up or down potential output port choice is disqualified if the row inhibit flag is true. Any right or left potential output port choice is disqualified if the column inhibit flag is true. When neither the row inhibit nor column inhibit flags are asserted, right or left is chosen over up or down, thus yielding a single direction choice.

Referring to Figure 8, each long distance messenger port input is connected through its control logic providing all of its lines except Req and Deny to all other port controls via distribution busses (808). Five such busses exist, one for each port control.

A control bus (806), containing signals from each port control carries control signals between ports as depicted in Figure 9.

Referring to Figure 9, each port control, using the control bus (806), is capable of asserting a request indication to any of the other ports. A port to which a request is asserted, using control bus (806), asserts a busy indication to all of the other ports. Thus, each port control receives four request indications and provides four busy indications.

Referring to Figure 7, each output port control (604) within the VFB contains a request resolution circuit (750) that provides a busy state of the output port to each of the input port controls within the VFB. When an output port is not is use, it provides a not busy indication to all of the input port controls.

Referring to Figure 8, an input port control (603), having selected a single direction choice, examines the busy status of the desired output port control (604) for the direction choice, as carried on the control bus (806) to determine if the port may be used. If the output port control for the chosen direction indicates busy, then the row or column inhibit flag of the input port is asserted to cause that direction of travel to be eliminated by the input port as a potential choice. If direction choice is up or down and the desired port is indicating busy, the row inhibit of the input port is asserted. If direction choice is right or left and the desired port is indicating busy, the column inhibit of the input port is asserted.

If the output port in the chosen direction does not indicate busy, then a request is asserted to that output port control (604). Upon receiving the request assertion, the output port control causes a busy indication to be asserted to all input ports controls except the input port asserting the request. This causes all other ports to be prevented from attempting to utilize that port for as long as the request is asserted. The potential exists for a very brief race condition within the request resolution circuit (750). If an output port control is not busy when an input port control has received a message request, and during the period between when the request is asserted to the output port control and the busy is propagating to the other input port controls, an additional message request requiring the same output port control may be received and asserted to the output port control. This results in multiple concurrent requests for the same output direction. The request resolution circuit then indicates, as a result of the multiple concurrent requests, a busy indication to the input port controls for all of the requesting directions, forcing each to select an alternate path.

Once the request resolution circuit (750) of an output port control (604) has resolved the incoming requests and single input port control thereby selected, , it then channels the data from the distribution bus (808) driven by the selecting input port control to the output port. The Data lines of the input port are passed unchanged to the output port. The address portion of the input port is passed through to the output port with the row or column offset decreased. For the up or down output port, the row portion of the address is decreased; for the right or left output port, the column portion of the address is decreased. This Address Decrease circuitry is shown as (751) in Figure 7. Thus, as the message request is moved nearer its destination through a VFB, its address offsets are adjusted appropriately so that upon arriving at the desired destination within the array of VFBs, the address offsets for both row and column are zero. The Strobe line is passed, unchanged, from the input port to the output port. The Grant and Abandon lines are passed, unchanged, from the output port to the input port, and finally, the Req line of the output port is asserted.

The Deny line is not passed directly from the output port to the input port. Instead, when asserted to the output port by the input port of an adjacent VFB, the port control output asserts a busy indication to the selecting input port control. The input port control then forces the row inhibit or column inhibit to be asserted. If the selected output port control is an up or down port, the row inhibit of the input port control is asserted. If the selected output port control is a left or right port, the column inhibit of the input port control is asserted.

In the event that the input port control lacks the availability of any port that would move the message towards its destination, it conveys this inability back to the output port of the adjacent VFB that is connected to the blocked input port by asserting a Deny signal. Deny is generated by the input port control when it detects that both the row inhibit and column inhibit flags are asserted and the row and/or column offset(s) is/are not zero. When Deny is received by an output port, it causes a busy signal to be generated to the input port control that has selected it. This in turn causes the either the row inhibit or column inhibit of the input port control to be asserted, resulting in the dropping of the request for the output port control by the input port control and the generation of a different choice of output port control, or if no choice remains, to generate a deny signal back to the messenger port feeding the input port control. Once connected to the destination VFB, the final input port choice must be the internal output port control. If the internal output port control is asserting a busy state to the selecting input port control, the input port control will assert Abandon rather than Deny to the input port, as there is no alternate path possible.

Thus, once a message request has been asserted to the long distance internal input port (314) of a VFB, the long distance messenger system will exhaustively attempt all potential routes to the destination VFB indicated in the address. Port selections that would result in routes that are circuitous, i.e. would not move a message closer to the destination with each segment constructed, are automatically excluded by the process described. This results in minimum orthogonal length path being constructed between a source and destination VFB through which the source may transmit data and control information to the destination. If a path could not be discovered, the long distance internal input port (314) will receive either a Deny or Abandon signal indicating the attempt was not successful.

Initial Connected State

In the event that a message path is successfully constructed to the destination, the Data lines on the internal output port (312) of the destination VFB – at the time the Req line is asserted on the internal output port – reflect the contents of the MSENDA register of the source VFB that originated the request. The long distance to local messenger bridge (311) in the destination VFB then latches the contents of the Data lines, feeding the Register Index, Force and Interrupt portion of the latched information to the local messenger input port (310) connected to the bridge. The source VFB is now in the initial Connected State.

Connected State

After the Initial Connection State has been established, the messenger bridge (311) in the destination bridge monitors the state of the presence bit for the messenger register specified by the register index value present at the input port (310). If the Present line of the input port indicates not asserted or the Force line of the input port is asserted, the messenger bridge logic asserts the Grant line to the internal output port (312), and forces the Deny and Abandon lines of the output port to be forced to a non-asserted state for as long as the Req line of the output continues to be asserted.

During the Connected State, the state of the Grant line on the output port (312) of the destination VFB is then propagated through the established path to the long distance internal input port (314) of the source VFB. When the Grant signal has propagated back to and is sensed by the source VFB, the source VFB is considered to be in the Connected State.

Connected Transfer State

During the Connected State, when the control logic (304) in the source VFB detects the assertion of Grant on the input port (314) and then detects the presence bit associated with the MDATAW register to be asserted, it enters the Connected Transfer State by causing the contents of the MDATAW register within its register file to be placed onto the Data lines of input port (314) of the source VFB, and causing the Strobe line of the port to be driven concurrently.

Next during the Connected Transfer State, the strobe and data present on the Data lines of the input port (314) within the source VFB are propagated through the established path to the output port (312) within the destination VFB. The messenger bridge (311) within the destination VFB, in response to detecting the assertion of the strobe line, transfers the data present on the lines of the output port (312) to the Data lines of the local input port (310) and asserts the Req line of the local input port, effectively generating a local messenger request with the destination VFB.

Next during the Connected Transfer State, the local request is processed by the local messenger (305) within the destination VFB as described previously, resulting in a constructive access to the messenger register (301) indicated by the local request on the input port (310). Upon conclusion of the local request, the Ack line of the local input port (310) is asserted, resulting in the de-assertion of the Req line of the local input port. As a result of this constructive access, the presence bit associated with the accessed register is asserted. This is detected by the messenger bridge (311) causing the Grant line of output port (312) to be de-asserted, and propagated through the established path from the destination VFB to the input port (314) of the source VFB.

Next during the Connected Transfer State, the control logic (304) of the source VFB, detecting the de-assertion of Grant on the input port (314), causes the presence bit associated with the MDATAW register to be de-asserted, and the Strobe line of input port (314) to be de-asserted. Concurrently, if the source VFB control logic (304) detects that the Burst Transfer bit of the MSENDA register is not asserted, the presence bit of the MSENDA register within the source VFB is de-asserted, and the control logic causes the Disconnecting State to be entered, otherwise the Connected State is re-entered.

Disconnecting State

If, at any time during any state other than the Disconnected State the control logic (304) within the source VFB causes the Req line of input port (314) to be de-asserted, the Disconnecting State is entered. During the Disconnecting State, each VFB that constitutes the path constructed between the source VFB and destination VFB passes forward the de-assertion of Req from the source VFB towards the destination VFB. The de-assertion of Req is passed along any VFB through which a path (either completed or partial) was established. When Req is de-asserted on the input port (314), its request for a corresponding output port (205, 206, 207 or 208) is de-asserted, causing the output port in turn to de-assert Req to its output port. Thus the destruction of the path occurs from source VFB to destination VFB as a result of the Req line of the input port (314) of the source VFB being de-asserted, and resulting in the Disconnected State being entered. The control logic (304) will de-assert the Req line to the input port (314) when it detects the presence bit associated with the MSENDA register to be no longer asserted.

Burst Transfers

After the first Connected Transfer State, if the Burst Transfer bit of the MSENDA register within the source VFB is asserted, the source processor (303) may continue to transfer data and control information to the destination VFB by performing repetitive constructive accesses to the MDATAW register. Each constructive access to the MDATAW register of the source VFB will cause a Connected Transfer State to be entered when the control logic (304) of the source VFB detects the assertion of the Grant line on the internal input (314). The source and destination VFB will remain in a Connected State until the processor (303) within the source VFB causes a destructive access to the MSENDA register.

Abandoned State

If, after entering the Initial Connected State, but before entering the Connected State, the messenger bridge (311) in the destination VFB when monitoring the state of the presence bit for the messenger register specified by the register index value present at the input port (310) determines the present bit of the input port is asserted and the Force bit of the internal output port (312) is not asserted, the messenger bridge logic asserts the Abandon line to the internal output port (312), and forces the Deny and Grant lines of the output port (312) to be forced to a non-asserted state for as long as the Req line of the output continues to be asserted.

During this abandoned state, the Abandon signal is propagated from the output port (312) of the destination VFB, to the input port (314) of the source VFB. Control logic (304) within the source VFB then causes the retry state to be entered.

Retry State

The Retry State causes the Disconnecting State, Disconnected State and, potentially, a Connecting State to be entered by forcing the Req line of the long distance messenger internal input (314) to be de-asserted by control logic (304) within the source VFB for an arbitrary delay period. This delay period is 8.33 nanoseconds in the preferred embodiment.

Deny State

If, during the Connecting State, but before entering the Connected State, the source VFB control logic (304) detects the assertion of the Deny signal on the input port (314), the Retry State is entered.

Overall Operation, Single Transfer, Without Path Contention

To further clarify the general operation, referring to Figure 10, the overall operation of sending a message from a Source VFB (1001) to general purpose messenger register M1 (register index 9 hex, see table 1) of the register file of the destination VFB (1002) consists of the following sequence. Row and column offset addresses include the state of the negative row and negative column bits, thus are summarized as {Row Offset, Column Offset} for this operational description.

First, the processor (303) of source VFB (1001) performs a constructive access to the MSENDA register of its messenger register file (301), with a value if 00010389 hex, indicating that register M1 of the VFB located 3 columns to the right, and 1 row above is to be modified {-1,3}. Next, processor (303) of source VFB (1001) performs a constructive access to the MDATAW register of its messenger register file (301), with an arbitrary value to be placed into the M1 register of the messenger register file (301) within the destination VFB (1002).

Next, the control logic (304) within the source VFB in conjunction with the optimizer (315) determines, based on the row and column offset addresses present in the source VFB MSENDA register, that long distance messenger is required to transfer to the data to the destination VFB (1002), and enters a Connecting State as described previously.

Next, the long distance messenger within the source VFB (1001) selects the right side output path – its preferred direction – and asserts a long distance request to the VFB to the right (1003). The right side long distance output of VFB (1001) is presenting row and column offset values of {-1, 2} to the left side input of VFB (1003).

Next, the long distance messenger of VFB (1003), selects the right side output path – its preferred direction – and asserts a long distance request to the VFB to its right

(1004). The right side long distance output of VFB (1003) is now presenting row and column offset values of {-1, 1} to the left side input of VFB (1004).

Next, the long distance messenger of VFB (1004), selects the right side output path – its preferred direction – and asserts a long distance request to the VFB to its right (1005) The right side long distance output of VFB (1004) is now presenting row and column offset values of {-1, 0} to the left side input of VFB (1005).

Next, the long distance messenger of VFB (1005), having its column inhibit flag asserted due to the zero column offset it is receiving from VFB (1004), selects the alternate direction, the up side output path, and asserts a long distance request to the destination VFB above (1002). The up side long distance output of VFB (1005) is now presenting row and column offset values of {0, 0} to the down side input of VFB (1002).

The long distance messenger of destination VFB (1002), recognizing the row and column address as {0, 0} received on its down side port, asserts a long distance request to its internal long distance messenger output port. The control logic of the destination VFB (1002), in turn feeding its long distance to local messenger bridge (311), in turn feeding the local messenger internal input port (310), achieves an Initial Connection State.

The messenger bridge of the destination VFB (311) – in conjunction with internal input port (310) and control logic (304) – examines the presence bit of the M1 register index described in the request. It determines that the transfer is not inhibited, thus enters a Connected State by asserting Grant to the internal input (314).

The source VFB (1001), reacting to the Grant assertion passed back from the destination VFB to the source VFB, enters a Transfer State that transfers the contents of the source VFB (1001) MDATAW register to the M1 register of the destination VFB (1002). Following the Transfer State, the source VFB (1001) returns to the Connected State, and, as indicated by a non-asserted Burst mode bit within its MSENDA register, enters a Disconnecting State followed by a disconnected state.

Upon conclusion of the message transfer, the source VFB (1001) has the presence bits associated with its MSENDA and MDATAW messenger registers not asserted, and the destination VFB (1002) has the presence bit associated with its M1 messenger register asserted and the M1 register contains the contents of the source VFB MDATAW register at the time of the Transfer State.

Overall Operation, Burst Transfer, Without Path Contention

Also referring to Figure 10, and using the same example as in Overall Operation, Single Transfer, Without Path Contention described above, but with the Burst mode bit asserted within the MSENDA register of the Source VFB (1001), the source may transfer multiple values to the destination VFB (1002).

After the first Transfer State has been performed and the source VFB has returned to the Connected State, the Disconnecting State is not entered. Instead, the source VFB (1001) remains in the Connected State. As a result of the initial Transfer State, the Grant line presented to the source VFB through the established path is now de-asserted, reflecting the assertion of the presence bit of the M1 messenger register of the destination due to the presence of data in that register.

The processor of the source VFB (1001) then performs additional constructive accesses to its MDATAW register, while the processor of the destination VFB (1002) performs destructive accesses to its M1 register, thus consuming the values transferred to it by the source VFB, until the number of values desired by the source VFB have been transferred. The source VFB (1001) then terminates the connection by performing a destructive access to its MSENDA register or causing the presence bit associated with its MSENDA register to be not asserted via the MSTATUS register.

During the process, each Transfer State is conditioned upon a value being present in the source VFB MDATAW register and a value not being present in the M1 register of the destination VFB (1002), as indicated by the state of the presence bits associated with each.

A constructive access by the processor of the source VFB (1001) to its MDATAW register when a Transfer State has not yet transferred the previous value results in the processor of the source VFB being forced to wait until the Transfer State is performed. Likewise, a passive or destructive access by the processor of the destination VFB (1002) to its M1 register prior to receiving a value through a Transfer State results in the processor of the destination VFB being forced to wait until the Transfer State is performed. Thus the source and destination processors operate in a synchronized manner through the established path.

Overall Operation, Retry of Communication

Also referring to Figure 10, and using the same example as in Overall Operation, Single Transfer, Without Path Contention described above, repeated attempts to transfer a value to the destination VFB (1001) will be performed if the messenger register within the destination VFB (1002) has its associated presence bit asserted at the time the Initial Connection State is reached.

After the source VFB (1001) has attained the Initial Connection State, if the presence bit associated with the messenger register within the destination VFB (1002) is asserted, the destination VFB will assert Abandon to the source VFB causing the source to enter the retry state. Consequently the path established is broken down then reestablished after the retry delay until a Transfer State is achieved.

Overall Operation, With Path Contention

Referring to Figure 11, and using the same example as in Overall Operation, Single Transfer, Without Path Contention described above, a portion of the path that would be utilized in the absence of contention is automatically circumvented. Presuming that a portion of an unrelated path has been established that consumes the long distance message path arriving at the down port of VFB (1004) feeding into the left port of VFB (1005), the operation varies as follows.

During the connecting state, the long distance messenger of VFB (1004), as a result of a busy asserted by the right side output path, will then select its alternate direction, up, and assert a long distance request to the VFB above (1006), with a decreased row address, now at zero, {0, 1}.

Next, the long distance messenger of VFB (1006), selects the right side output path – its preferred direction – and asserts a long distance request to the destination VFB (1002) , with a decreased column address, now at zero {0, 0}.

Thus an alternate path to the destination VFB was discovered in the presence of contention for a portion of the path between a source and destination.

Overall Operation, With Path Contention Resulting in Deny

Referring to Figure 12, and using the same example as in Overall Operation, Single Transfer, Without Path Contention described above, a portion of the path that would be utilized in the absence of contention is circumvented. Presuming that unrelated paths have been established that consume the long distance message path passing into the up port of VFB (1006) into the left port of VFB (1005), and passing into the down port of VFB (1004) and out of the up port of VFB (1006), the operation varies as follows.

During the connecting state, the long distance messenger of VFB (1004), as a result of a busy asserted by the right side output path, will then select its alternate direction, up. As a result of a busy asserted by the up side output path, there is no selection remaining that moves the path closer to the destination VFB (1002). VFB (1004), having no suitable direction choice available, asserts a Deny to its path predecessor, VFB (1003).

As a result of the Deny signal from VFB (1004), VFB (1003) causes the column inhibit for its left port to be asserted and the alternate path choice, up, to be selected. This succeeds, causing the selection of the up side output port of VFB (1003) to assert a long distance request to VFB (1007), with a decreased row address, now at zero, {0, 2}.

Next, the long distance messenger of VFB (1007), selects the right side output path and asserts a long distance request to VFB (1006), with a decreased column address, {0, 1}.

Next, the long distance messenger of VFB (1006), selects the right side output path and asserts a long distance request to the destination VFB (1002), with a decreased column address, {0, 0}.

Thus an alternate path to the destination VFB was discovered in the presence of contention for a portion of the path that required a portion of the route being established to be undone and an alternate path discovered in it place.

Peripheral Array Connections

Referring to Figure 1, the periphery of the array requires a stub (103) or input/output connection (104) to correctly terminate the local and long distance messenger connections.

A stub (103) requires no additional logic, but connects all of the local messenger input lines feeding into the VFB to a de-asserted state.

In addition, the stub connects all of the long distance messenger input lines to a de-asserted state with the exception of the Abandon and Grant lines of the output port, which should both be tied directly to the Req line of the output port. This will cause a unique state to be signaled to a source VFB, specifically the simultaneous occurrence of both Grant and Abandon during the Connecting State that the source VFB may use to detect a long distance message request to non-existent VFB.

A input/output connection (104) is used to transfer data into or out of the processing array (100), and may consist of all or any of the local messenger input port, local messenger output port, long distance messenger input port, or long distance messenger output port connected adjacently to the VFB to be accessible via external pins or other connection means to arbitrary external circuits. Any port not utilized must be terminated as described for the stub function (103).

Alternate Embodiments

The description set forth above is for the purposes of illustration. Not all of the features listed above are required to establish the scope of the invention. In additional, alternate embodiments and features are possible.

For example, the nomenclature, polarity, bus-widths, bit and word positions, and signaling levels as described in this disclosure are representative only and should not be viewed as limiting. In addition, while defined herein as electrical in nature, other means for implementing the current invention such as optical, mechanical, and chemical systems are to be considered as within the scope of this disclosure.

The functionality, nomenclature, and number of Processing Elements (referred to herein as Virtual Function Blocks or VFBs) to communicate via the present invention are outside of the scope of this disclosure; it is the means and apparatus used for dynamically discovering and creating/relinquishing these communications channels that is the focus of this disclosure. Similarly, while the description of the preferred embodiment describes an exemplary array of homogeneous processing elements, the current invention can be equally employed to connect heterogeneous elements. To be used in a system connected by the current invention, an element need only the ability to access the messenger register file. Further, the array of elements need not square; any combination of rows and columns can be accommodated simply by adjusting the usable range of bits allowed in the absolute Row Offset and Column Offset fields of the MSENDA register of the messenger register file. A maximum of a 255 by 255 VFB array is supported by the description in the preferred embodiment (eight bits defining absolute offset and one bit defining the direction for each of the two generalized dimensions). Larger arrays could be accommodated by extending the absolute offset bit fields.

This document describes an embodiment of the invention in terms of a single, integrated circuit, realized in a single piece of silicon. One benefit of such an embodiment is that functions connected by the invention only need to incorporate the described interface and control circuitry to allow seamless operation and communication within an array of functions with like interface and control circuitry. Therefore, the concepts and operation of the current invention can be extended to individual blocks or "tiles", placed together in an array such as in a multi-chip module (MCM) or even through separate, individually packaged devices connected externally, for example on a printed circuit board. No external interface or glue logic is required to connect such blocks; they simply need to be placed next to each other and wired together. All such physical embodiments are to be considered as within the scope of this disclosure.

Alternate hardware and/or software-based retry scenarios are possible and should be considered within the scope of this disclosure. Specifically, random, increasing delay, decreasing delay, and/or any combination of these retry scenarios are within the scope of this disclosure.

The interconnect mechanism disclosed herein for an exemplary two-dimensional array of VFBs can be extended to three or more dimensions. For each additional dimension, the interconnect and control structures – including extensions to the messenger register file and the arbitration / selection circuitry – would be replicated. For example, a three-dimensional array would add "above/below" to the "left/right" and "up/down" generalized dimensions of the current invention. Local message registers for data and address would be added in the messenger register file to support the ABOVE and BELOW directions, and the presence bits of such registers reflected in the MPRES and MSTATUS registers. The MSENDA register in messenger register file would require additional bit fields for describing the above/below absolute offset and direction. The local and long distance input and output message ports, plus the control logic driving these ports, would also be replicated in each VFB. The local messenger prioritizer, source selector, and control logic would have two additional sets of inputs, one from the "above" and one from the "below" local message ports. Request resolver, multiplexer control, multiplexer, and address decrease circuitry would be incorporated in each VFB for each new port (above and below, in this instance) and an additional distribution bus for each additional port would be connected to the port control logic of all other ports (including the internal port control). An inhibit flag – similar to the row and column inhibits of the current invention – would be required for the new dimension, and the Direction Selection circuitry would be modified to include the new dimension in the decision process. The concept can be extended to four or more dimensions by adding the necessary circuitry and interconnect for each new dimension.

As to a further discussion of the manner of usage and operation of the present invention, the same should be apparent to one skilled in the art from the above description.

With respect to the above description then, it is to be realized that the optimum dimensional relationships for the parts of the invention, to include variations in size, materials, shape, form, function and manner of operation, assembly and use, are deemed readily apparent and obvious to one skilled in the art, and all equivalent relationships to those illustrated in the drawings and described in the specification are intended to be encompassed by the present invention.

Therefore, the examples and embodiments described herein should be considered as illustrative only of the principles of the invention. Further, since numerous modifications and changes will readily occur to those skilled in the art, it is not desired to limit the invention to the exact construction and operation shown and described, and accordingly, all suitable modifications and equivalents may be resorted to, falling within the scope of the invention.

Docket No. GCTI-MESS1

ABSTRACT

A modular processor architecture with a self-routing, message-based interconnect system for electrical construction provides a flexible, efficient, self-routing and real-time dynamically optimized means for connecting together functional elements or blocks within a semiconductor device and/or other electrical or electronic system. The device allows a homogeneous or heterogeneous array of semiconductor-based functional blocks to be connected with no external control logic. Such functional blocks, when connected via the current invention as described herein, discover, use, and then release dedicated communications channels between single or multiple sources and destinations through the array, with blocks between any given source and destination participating in finding and establishing the communication channel and then passing control and data information through from source to the destination.. Multiple communications paths can be automatically and simultaneously discovered and used in real-time on an as-needed basis. The message handling circuitry is completely combinatorial and not dependant upon any system or local clock. Ergo, the combinatorial path-discovery circuitry interrogates potential minimal-length communication paths – exhaustively if necessary – and establishes the first available path at speeds limited only by the intrinsic delays of the physical silicon in which the invention is implemented.

The functional blocks communicate to the present invention through a set of memory-mapped registers (Messenger Register file) and to other functional blocks via the present invention. The combination of a functional block and the current invention creates what is referred to herein as a Virtual Function Block or VFB. Systems built from VFBs can be constructed by physically aligning the VFBs and connecting them via metallization; no additional, external control logic is required. The metal (or other signal conductors) between the VFBs create the communication bus segments that form the paths for inter-VFB communications.

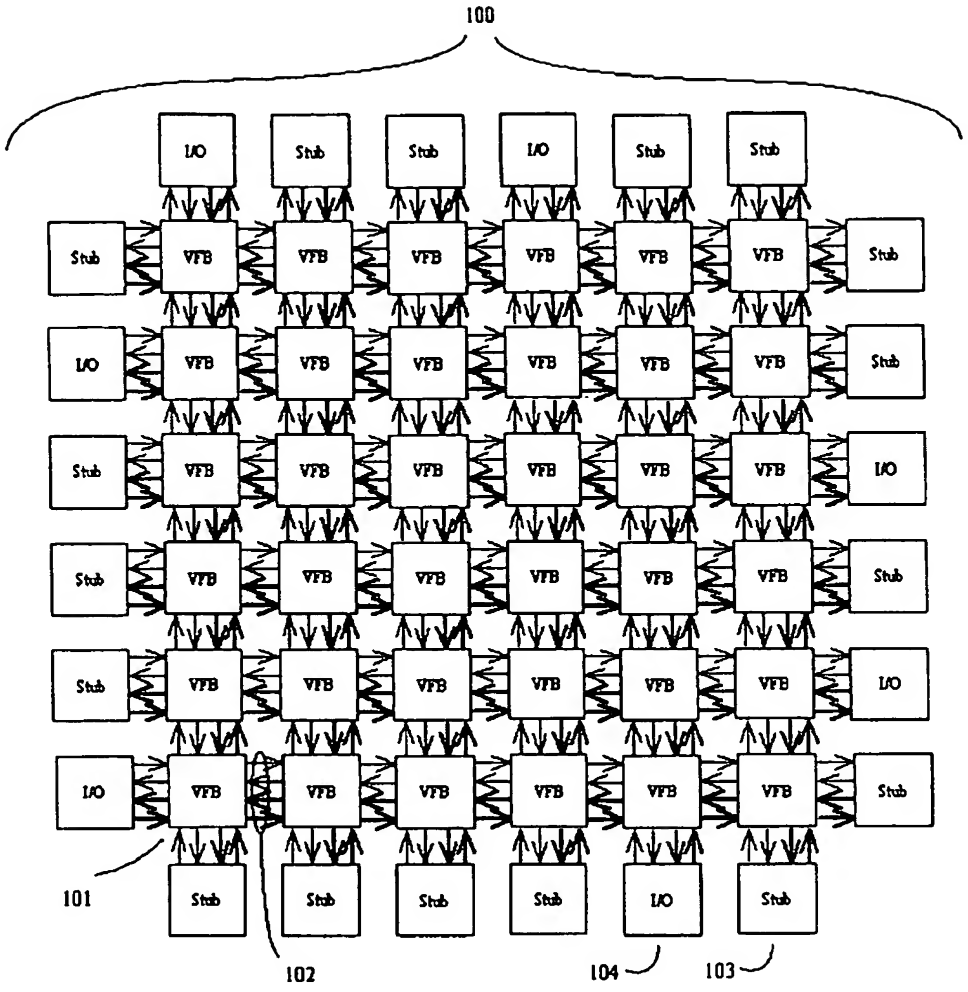


Figure 1. Messenger Interconnected Processor Array

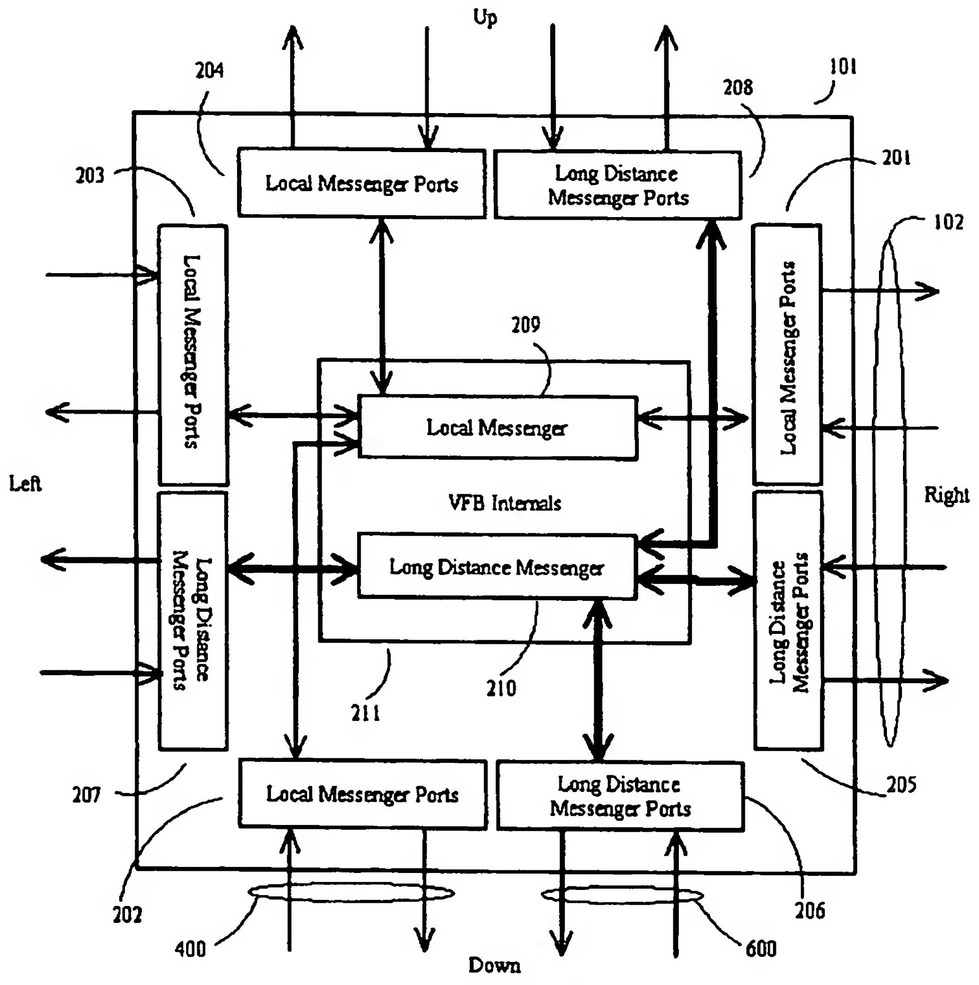


Figure 2. Individual VFB

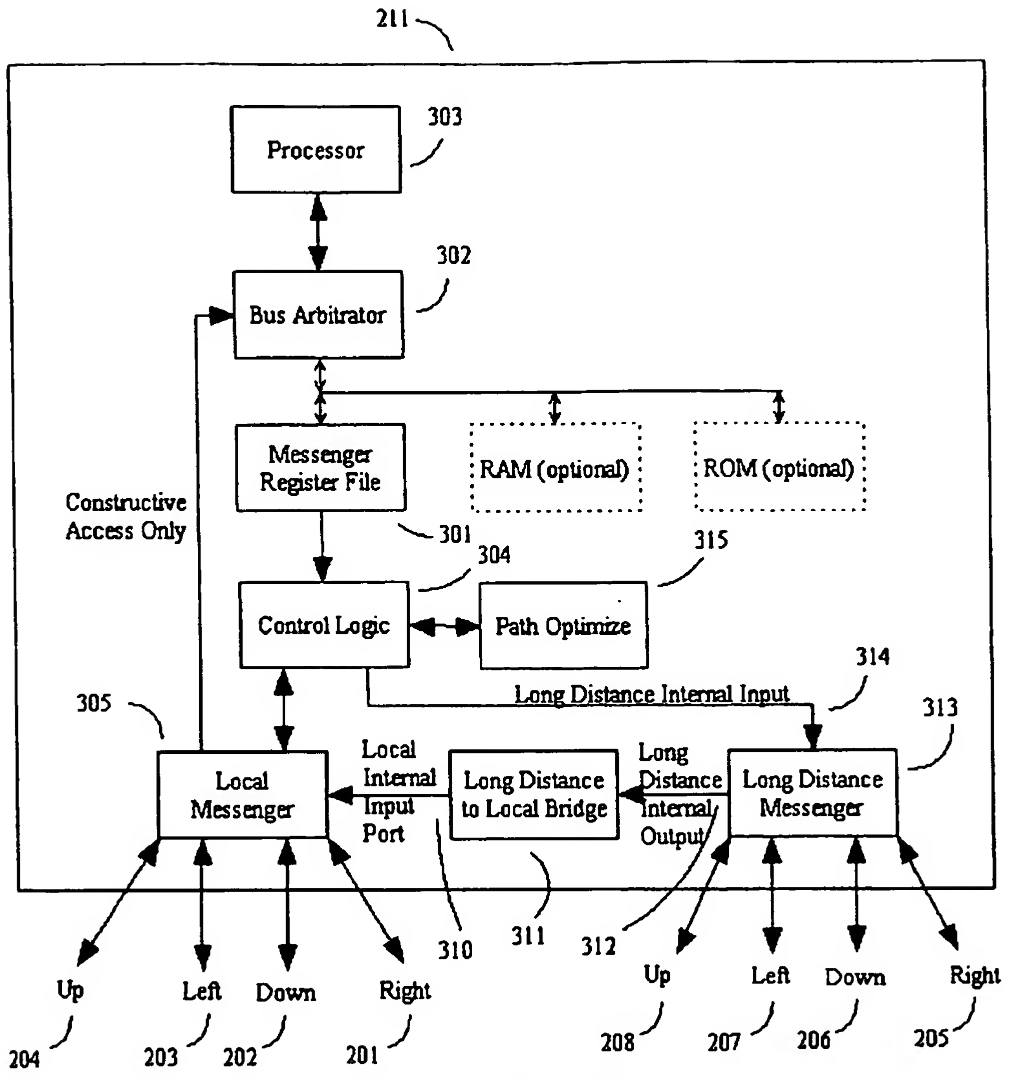
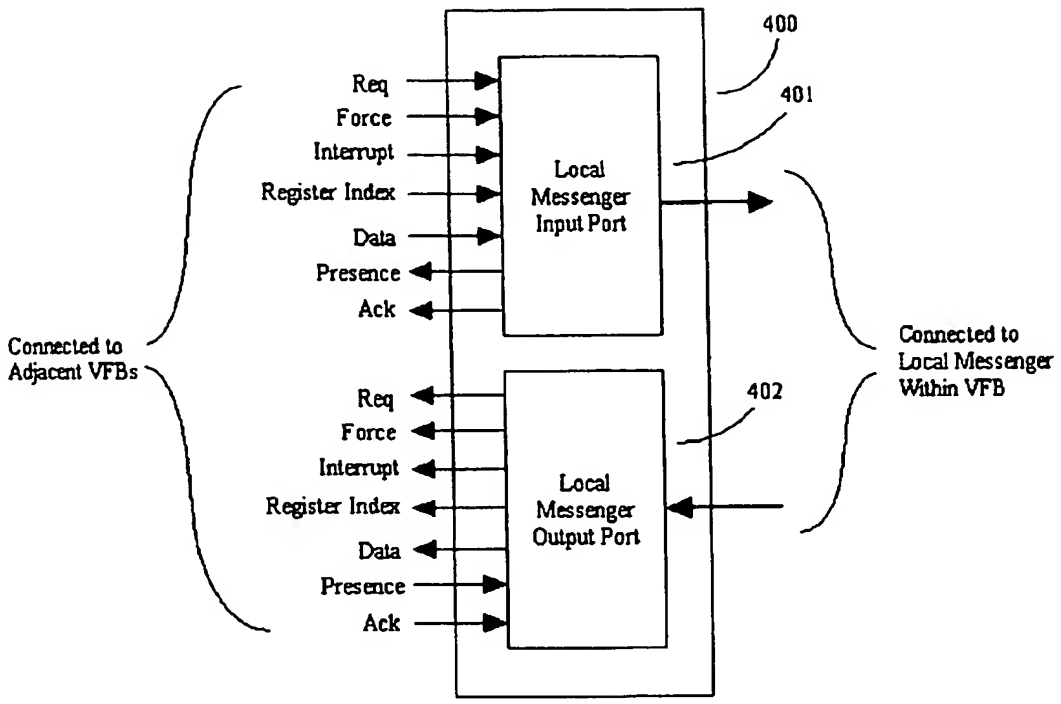
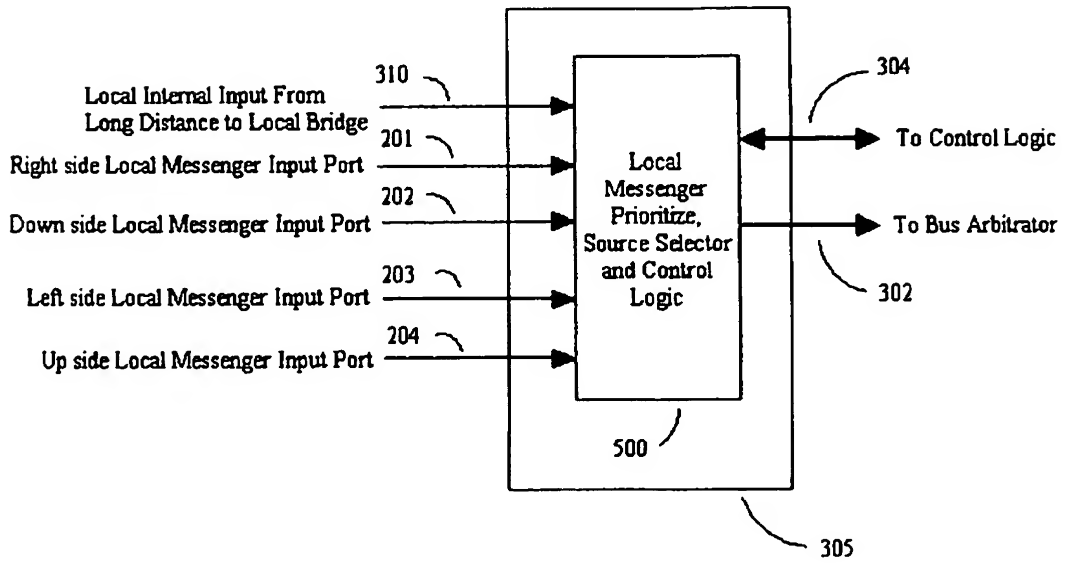


Figure 3. VFB Internals



400
401
Req
Force
Interrupt
Register Index
Data
Presence
Ack
Local Messenger Input Port
Connected to Adjacent VFBs
Connected to Local Messenger Within VFB
402
Req
Force
Interrupt
Register Index
Data
Presence
Ack
Local Messenger Output Port

Figure 4. Local Messenger Port



310
Local Internal Input From Long Distance to Local Bridge
201
Right side Local Messenger Input Port
202
Down side Local Messenger Input Port
203
Left side Local Messenger Input Port
204
Up side Local Messenger Input Port
Local Messenger Prioritize, Source Selector and Control Logic
304
To Control Logic
To Bus Arbitrator
302
500
305

Figure 5. Local Messenger

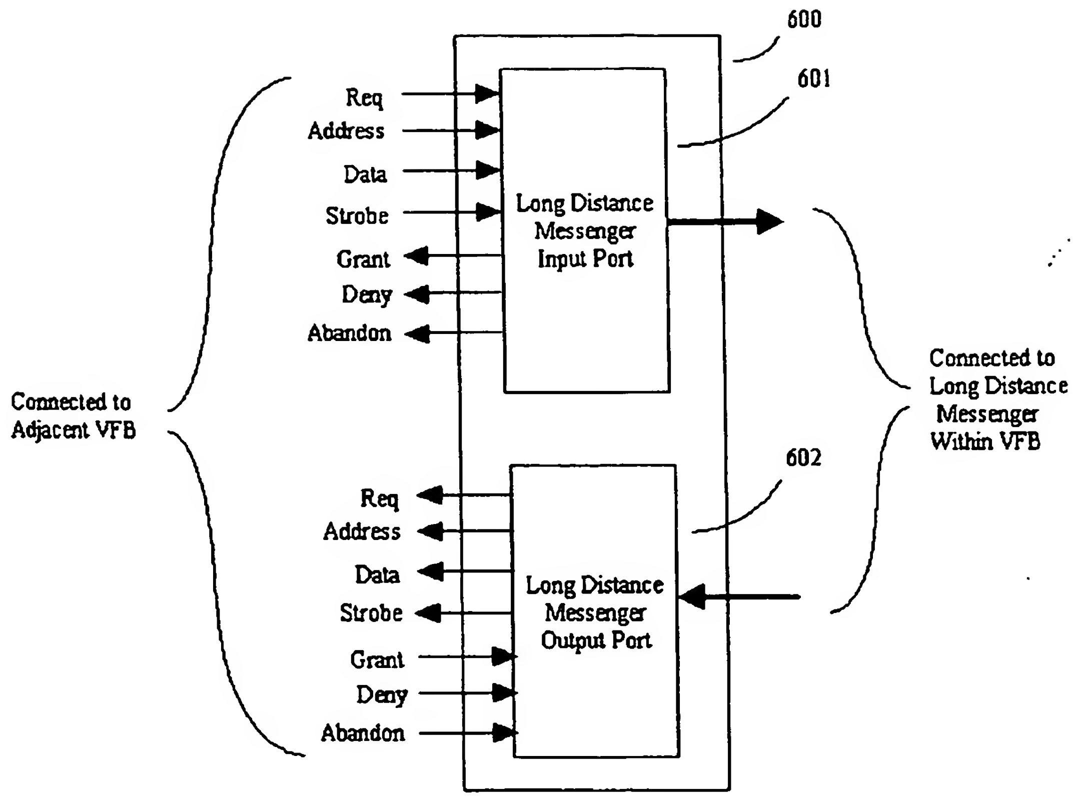
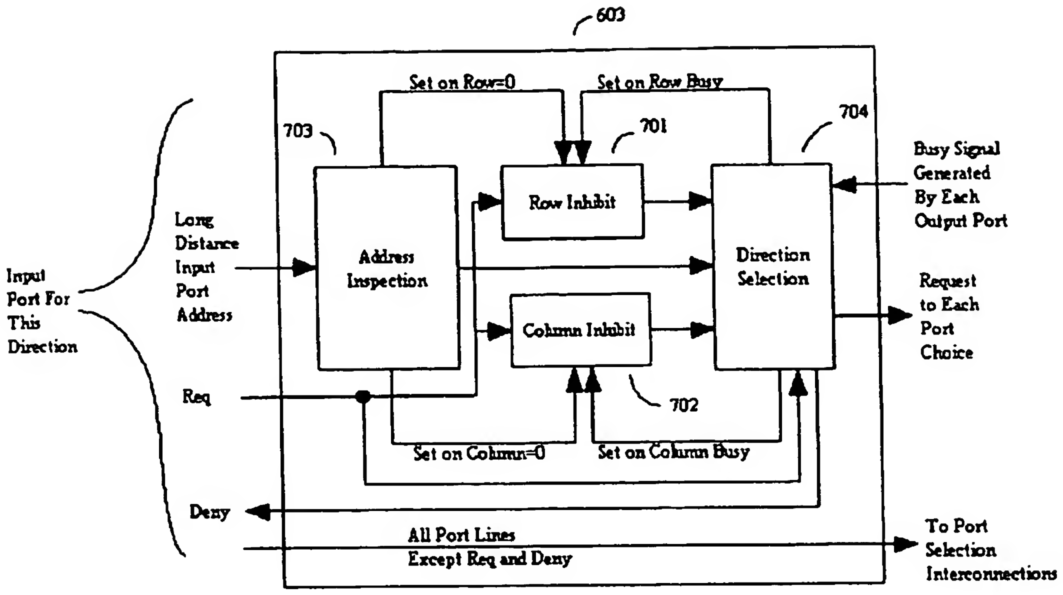

600
601
Req
Address
Data
Strobe
Grant
Deny
Abandon
Long Distance Messenger Input Port
Connected to Adjacent VFB
Connected to Long Distance Messenger Within VFB
602
Req
Address
Data
Strobe
Grant
Deny
Abandon
Long Distance Messenger Output Port

Figure 6. Long Distance Messenger Port


603
Set on Row=0
Set on Row Busy
703
701
704
Row Inhibit
Busy Signal Generated By Each Output Port
Long Distance Input Port Address
Address Inspection
Direction Selection
Input Port For This Direction
Request to Each Port Choice
Column Inhibit
Req
702
Set on Column=0
Set on Column Busy
Deny
All Port Lines Except Req and Deny
To Port Selection Interconnections

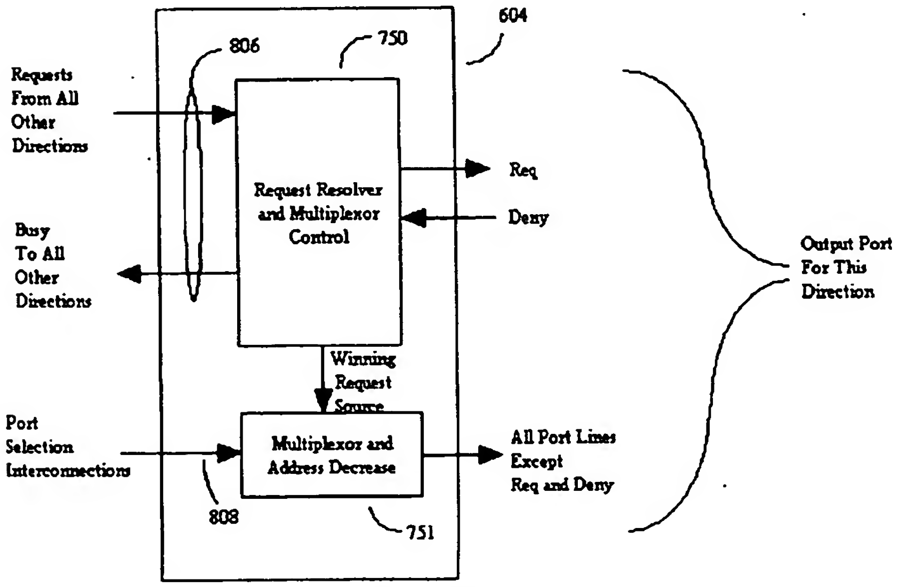
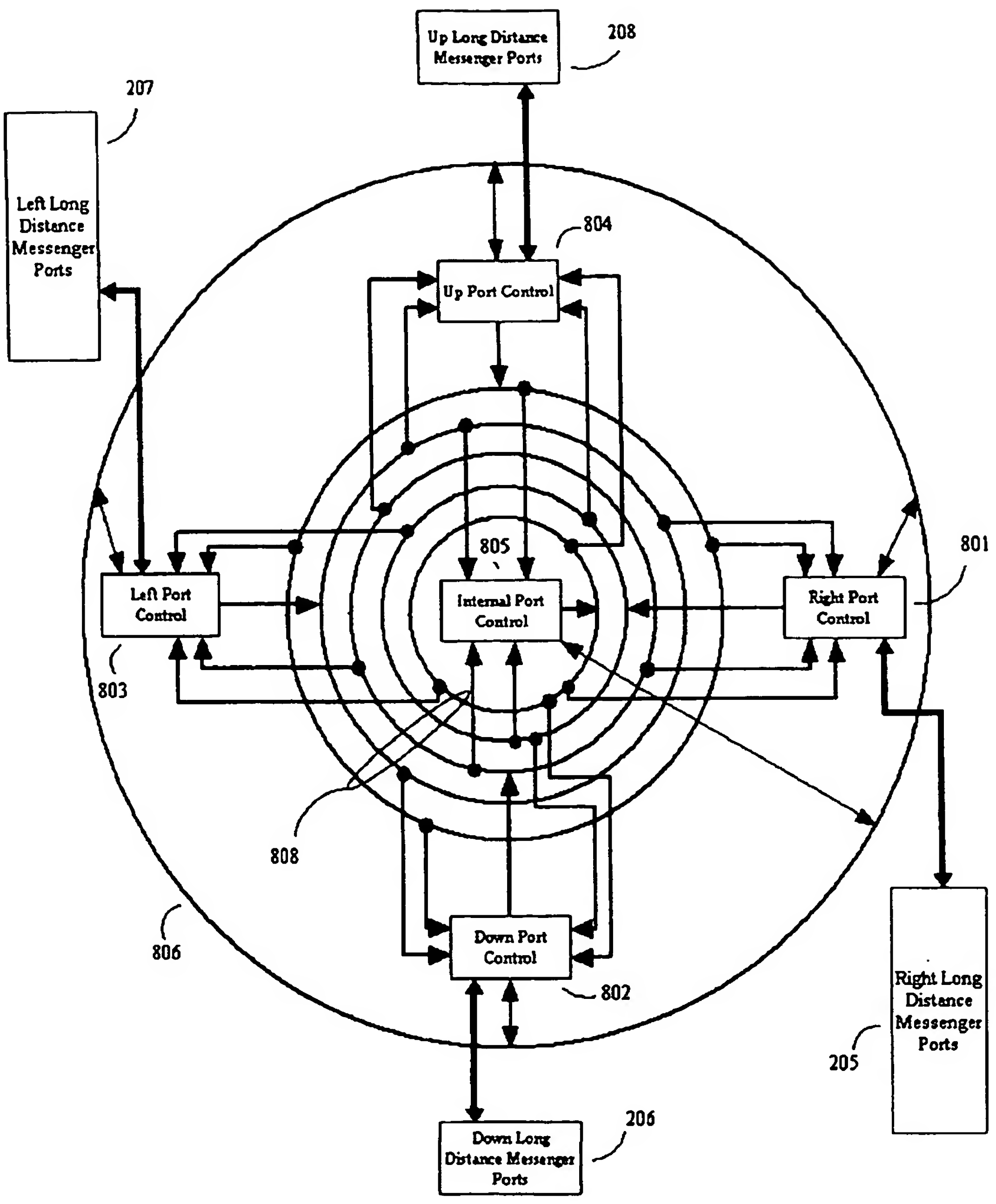

604
806
750
Requests From All Other Directions
Request Resolver and Multiplexor Control
Req
Deny
Busy To All Other Directions
Output Port For This Direction
Winning Request Source
Port Selection Interconnections
Multiplexor and Address Decrease
All Port Lines Except Req and Deny
808
751

Figure 7. Long Distance Messenger Port Control Logic



Up Long Distance Messenger Ports
208
207
Left Long Distance Messenger Ports
804
Up Port Control
805
Internal Port Control
Left Port Control
Right Port Control
801
803
808
806
Down Port Control
802
Right Long Distance Messenger Ports
205
206
Down Long Distance Messenger Ports

Figure 8. Long Distance Messenger

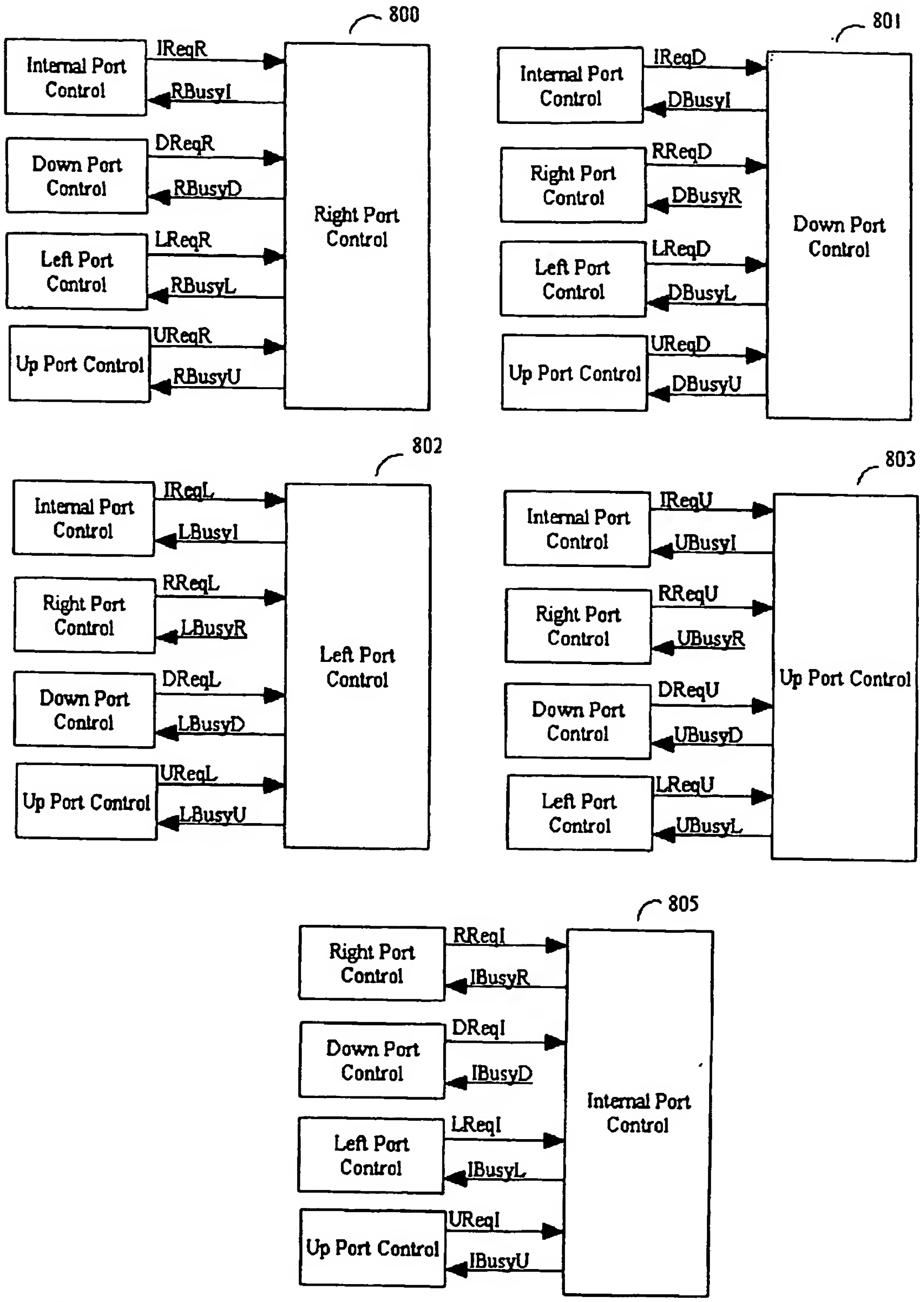


Figure 9. Request and Busy Information Shared Among Long Distance Port Controls

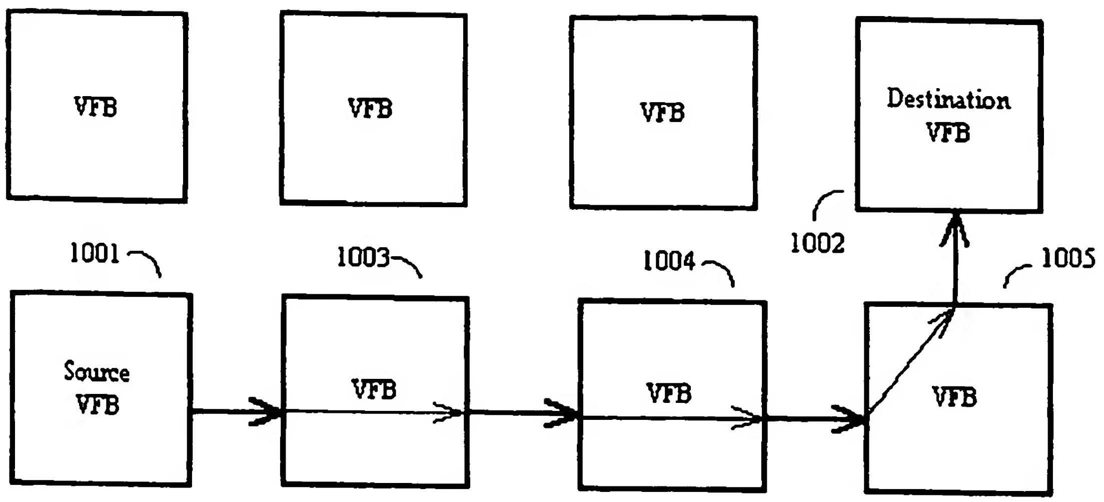


Figure 10. A Long-Distance Message Connection, No Path Contention

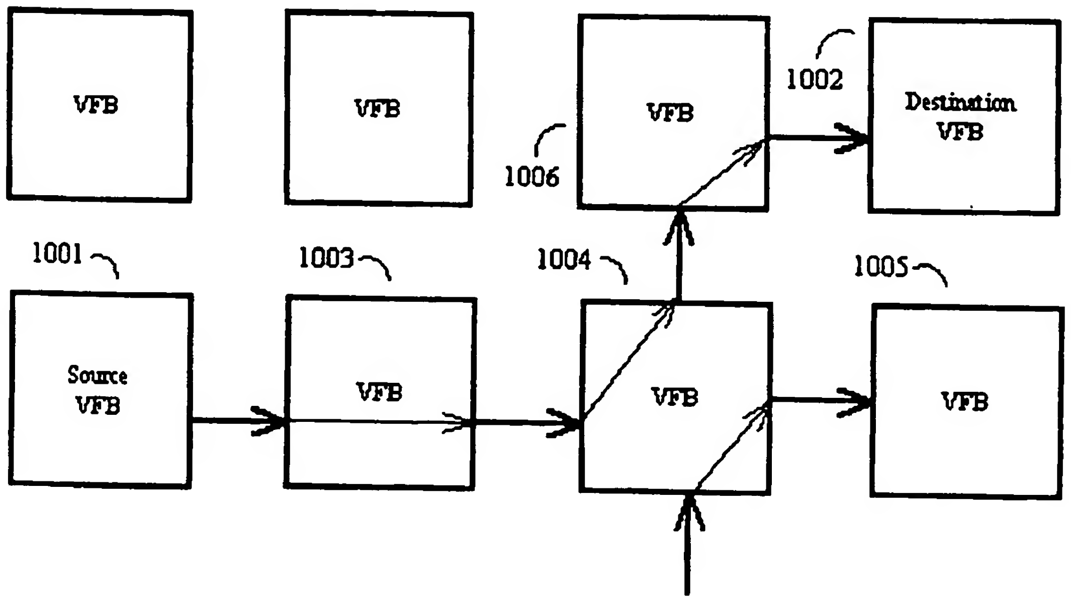


Figure 11. A Long Distance Message Connection, Path Contention Without Deny

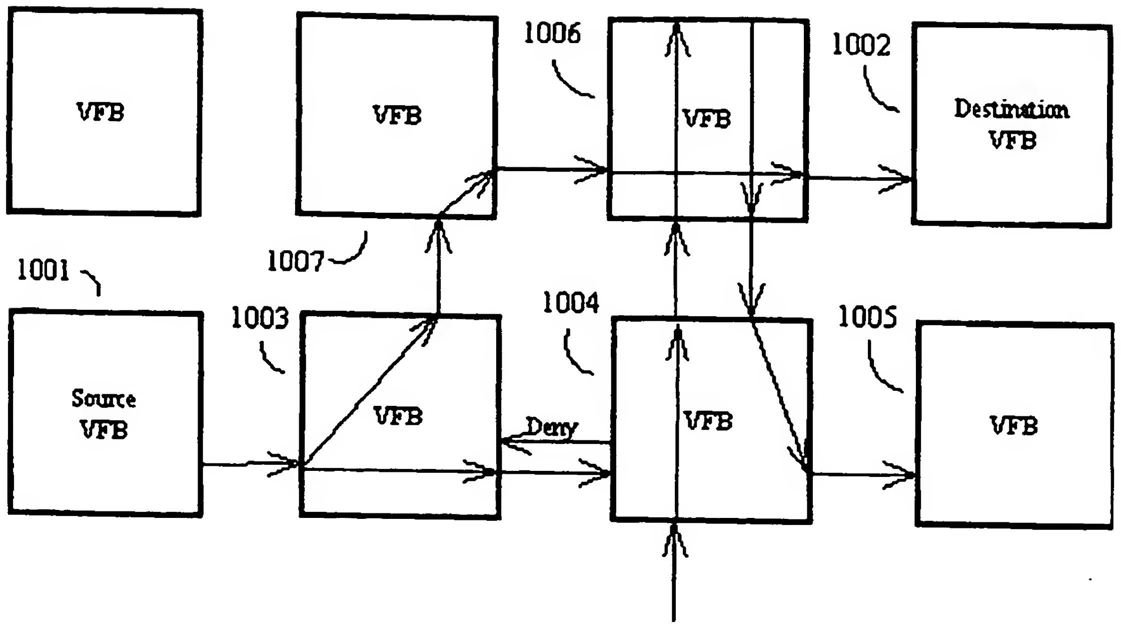


Figure 12. A Long Distance Message Connection, Path Contention With Deny